POLITECNICO DI MILANO
Master of Science in Computer Science Engineering
Department of Electronics, Information and Bioengineering

Attention Methods in remote sensing scene classification: the case of illegal landfills

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Supervisor: Piero Fraternali
Co-supervisor: Rocio Nahime Torres

Master Thesis of:
Andrea Biscontini, 901310

Academic Year 2020-2021

Abstract

Illegal landfills have become one of the most profitable businesses for criminal organizations and an increasing burden on the economy, the environment and, above all, the health of citizens. In the fight against this phenomenon, in order to identify these sites at an early stage and prevent damage, ongoing research is focusing on automating the process of illegal dumps detection. Among them, one of the more modern approaches is the use of Deep Learning models based on Convolutional Neural Networks (CNNs), that could enable mass-scale territory monitoring campaigns. In particular, using the ResNet50 architecture has already delivered good results.

This thesis evaluates the effects of adding Attention mechanisms to the above mentioned network. These are techniques designed to enhance a CNN by focusing its computational resources on the most significant parts of the input data. Their application on this task of illegal dumps detection could be useful for a more precise identification of the single wastes present in the images. In particular, between these mechanisms, the Squeeze-and-Excitation (SE), the Convolutional Block Attention module (CBAM) and the Efficient Channel Attention (ECA) have been implemented on top of the existing ResNet50 architecture. Several configurations of these models were tested, and a quantitative and qualitative evaluation showed ECA to be the best option, obtaining the larger improvement on the classification performance.

Finally, the Class Activation Maps (CAMs), that are heatmaps designed to highlight the areas in an image that contributes the most to the classification, have been analyzed, to better understand the various models' capacity to identify the relevant objects in an illegal landfill scene. An ad-hoc dataset has been created with annotations of the relevant waste objects, and a quantitative analysis has been made comparing the CAMs with these ground truths. The results proved that the CAMs could be the first step towards weakly-supervised object detection and that, consistently with the previous results, ECA is the most effective attention module.

Sommario

Le discariche illegali sono diventate uno dei business più redditizi per le mafie, e un peso via via crescente per l'economia, l'ambiente e sopratutto la salute dei cittadini. Al fine di identificarne tempestivamente questi siti ed evitarne i danni, molte ricerche si sono concentrate sull'automazione del processo di rilevamento preventivo. Tra questi, uno degli approcci più moderni è l'uso di modelli di apprendimento profondo basati su Reti Neurali Convoluzionali (CNN), che potrebbero consentire campagne di monitoraggio del territorio su larga scala. In particolare, l'utilizzo dell'architettura ResNet50 ha già dato buoni risultati.

Questa tesi si propone di valutare gli effetti dell'aggiunta di metodi di attenzione alla suddetta rete già adoperata per questo task. Questi ultimi consistono in tecniche per potenziare una CNN, concentrando le risorse computazionali sulle parti più significative dei dati in input. Il loro impiego nel rilevamento di discariche illegali potrebbe essere utile per un'identificazione più precisa dei singoli rifiuti presenti nelle immagini. Tra questi meccanismi, lo Squeeze-and-Excitation (SE), il Convolutional Block Attention Module (CBAM) e l'Efficient Channel Attention (ECA) sono stati implementati sull'architettura esistente di ResNet50. Di questi sono state sperimentate molte configurazioni, ed è stata eseguita una valutazione quantitativa e qualitativa che ha dimostrato come ECA sia la migliore opzione, in grado di migliorare le prestazioni di classificazione.

Infine sono state analizzate le Class Activation Maps (CAMs), che sono heatmaps atte ad evidenziare le aree di un'immagine che determinano maggiormente la classificazione, al fine di comprendere meglio la capacità dei vari modelli nell'identificare gli oggetti rilevanti nelle scene contenenti discariche abusive. Un dataset ad-hoc è stato creato annotando manualmente i rifiuti presenti nelle immagini ed un'analisi quantitativa è stata fatta confrontando le CAMs con queste annotazioni. I risultati hanno mostrato che le CAMs potrebbero essere utili per effettuare rilevamento degli oggetti semi-supervisionato, e che ECA è il modello di attenzione migliore.

Acknowledgements

First and foremost I would like to thank my parents, Michela and Giovanni, who have supported me throughout all my studies, allowing me to find and cultivate my interests.
Moreover, I'm really thankful to Professor Piero Fraternali, who gave me the opportunity to work on this thesis project and have a brief experience in the world of academia and research, despite this difficult period of the pandemic.
During this months, the person who followed me most closely and to whom I owe my warmest thanks is PhD Rocio Nahime Torres, for everything she taught me and for her patience.
A final thanks goes to Benedetta, to all my friends that supported me, and to all the ones that i met during the times at Politecnico.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Waste disposal is one of the biggest challenges of the modern century. It is estimated that globally, in 2016, 2.01 billion tonnes of waste were generated, and by 2050 waste generation across the world is expected to reach 3.40 billion tonnes [30]. It is also estimated that the total cost to manage the disposal range from $205 billion in 2012 to the projected $375 billion in 2025 [22]. In this global context and market, various forms of criminal organizations all around the globe developed their interests and illegal traffics. In 2016, Interpol and UN Environment estimated that environmental crime is currently the fourth most lucrative illegal business globally, amounting to an annual cost that is estimated in a range between $91 and $258 billion every year [24]. Although estimates differ and are very broad, environmental crimes are generally considered the fourth largest criminal revenue stream, after drug trafficking, counterfeit crimes, and human trafficking [16].

This leads to a huge threat to public safety and health, and to the environment. **Illegal landfills** are unauthorized waste deposits, ranging from small dumps of citizens' garbage to wide landfills with buried dangerous substances managed by criminal organizations. These are especially dangerous because criminals usually set them on fire to cover up the evidence, leading to toxic fumes like dioxins released into the air. Furthermore, infiltration into the soil and into water sources is usually a consequence of mismanaged landfills, which leads to additional long-term problems. This impact on the pollution of the air, the soil, and the water sources, damages agricultural areas and animals alike, leading to the spread and rise of diseases and respiratory problems in many areas [35].

In particular in Italy, these topics began to gain public interest in the 80s and 90s. Starting in 1980, new environmental organizations like *Legambiente* were born [33], with the aim of raising awareness on these topics in addition to the environmental, social, and public health threats they represent. Significant legislative progress was achieved with the creation of the *Commissione parlamentare d'inchiesta sul ciclo dei rifiuti* in 1995 [10], a

Figure 1.1: Terra dei fuochi

commission of the Italian government with the explicit goal of monitoring and tackling this illegal traffic. These problems, however, did not disappear, and soon after the public opinion has again been moved a lot in relation to scandals like *Terra dei fuochi*. The event was the discovery of a large area located in southern Italy, characterized by the burial of toxic waste, the presence of numerous illegal landfills scattered throughout the territory, and the ignition of numerous waste fires. The presence of those illegal wastes was correlated with a significant increase in the incidence of specific diseases, like a spike in mortality from leukemia and other cancers in the local population. From the latest report, *Ecomafia 2020* of *Legambiente* [34], very worrying data are emerging. The ascertained environmental crimes in 2019 are 34,648, +23.1% compared to 2018. The potential business of *ecomafie* is estimated to be worth €19.1 billion in 2019 alone.

In recent years, the situation in Lombardy has worsened dramatically. Between 2017 and 2019, 56 illegal waste deposits burned [2]. Many abandoned warehouses and industrial sites were reconverted to illegal waste dumps and disposal centers by the *Ecomafie* working in the territory. This led the Lombardy Region in last years to pay €25.9 million to reclaim 16 illegal waste sites [2].

To act against those Ecomafie, in addition to a tougher legislative frame-

work and an increase in criminal penalties for this type of crime, better methods to find and characterize the volume and content of the illegal sites need to be developed. The processes for illegal dumps detection are costly and time-consuming, which prevented proactive actions to anticipate environmental hazards, such as fire generation or water source pollution. The main bottleneck of this process is that experts need to manually check the photos to spot a potential landfill, thus not allowing for timely mass-scale territory monitoring campaigns to be conducted. Various studies focused on developing methods to support the automation of this task. With the use of **Remote Sensing** and **Geographical Information Systems (GIS)** it has been possible to create models able to detect possible candidate sites for illegal landfills, but nonetheless historical aerial images were still needed for final human validation. **Deep Learning** is the next area of methods to be a logical and promising candidate to tackle this problem. Following the great results that were recently achieved in various Computer Visions tasks, such as image classification, detection, localization, and segmentation, the also increasing quality of satellite images data available, and the willingness from the legislators to use Geospatial Intelligence as a new form of environmental compliance control [11], solidified the efforts on this approach. Still, it is not an easy task, given how waste dumps present themselves from aerial images. An example case is when an abandoned industrial site is filled with garbage, which appears from an aerial view as spilling over the building's boundary, with the area around containing clues such as scattered wastes, pallets, and so on. Also, many other factors could contribute to the detection of an illegal landfill, such as the isolation of the area, trucks presence, stressed vegetation, and easy access to the site through roads. For all these reasons the problem of detecting the location of waste dumps in aerial images can be seen as a scene classification task. With scene classification we are referring to problems where the images are analyzed and classified based on the presence of different objects and the spatial relationships between them [38].

Following the previous reasoning, the Lombardy Region decided to launch in March 2019 an experimental project called ***Savager*** (Sorveglianza Avanzata Gestione Rifiuti). This project have been developed by ARPA (Agenzia Regionale per la Protezione dell'Ambiente) in collaboration with the Department of Electronics, Information and Bioengineering (DEIB) of the Politecnico di Milano, and in particular with a group of researchers coordinated by Professor Piero Fraternali. The aim is to introduce the technologies of Geospatial Intelligence and Earth observation by satellite, airplane, and drone, for the environmental monitoring on the regional territory. Starting with high-definition satellite images from Google Earth and also (starting from October 2020) from the World View 3 satellite, ARPA experts carry out sweeping surveillance, aided by reports from a Deep Neural Network developed by researchers at the Politecnico and trained specifically to classify

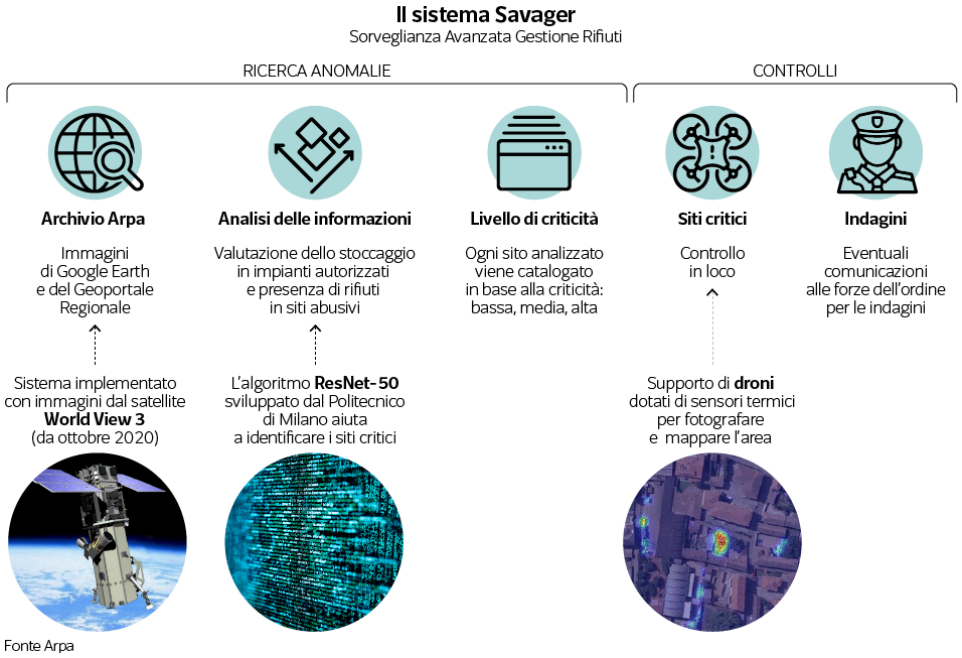


Figure 1.2: Savager Pipeline [2]

suspected illegal sites. Following a process of analysis and assessment of the critical issues of reported areas, the periodic surveillance process produces a refined list of suspect sites as an output. These are then subjected to on-site checks using technologically advanced drones, which are able to photograph and map the entire suspected area. Finally, if the findings justify it, the sites are communicated to the police for further investigations. A brief summary of the pipeline of this system is shown in Figure 1.2, taken from the blog of an Italian newspaper [2]. As of March 2021, 23 sites with serious problems were checked on-site and in 22 cases the presence of environmental offenses in waste management was confirmed. In 12 cases, the areas were also seized.

In the framework of this project, the Deep Neural Network that has been implemented by the DEIB researchers is the ResNet50 architecture. Given this initial model, another class of interesting and fairly new techniques that have been successfully applied to Convolutional Neural Networks are the **Attention mechanisms**. Their aim is to focus the computational resources on the smallest and most informative parts of the input data. These approaches lead to improvements in the prediction capabilities of various models in different tasks and, with the implementation of such mechanism, our aim was to check whether they could be beneficial also for this project. Between the various attention mechanisms, we have chosen to study and implement the following three: **Squeeze-and-Excitation Networks** (SE-Net [23]), **Efficient Channel Attention** (ECA-Net [58]), and **Convolutional Block**

Attention Module (CBAM [60]). Hence, the aim of this project is, using the already developed CNN with the ResNet50 architecture as the baseline, implement these selected attention modules (and some variations of them) over the ResNet50 backbone, and compare their results on the task of illegal landfills identification from aerial images. More specifically, the models were trained over a preexisting dataset of 2993 images to tackle a binary classification problem, the separation of the images in two classes, the first one being the positive (representing the presence of an Illegal Landfill in the image) and the second one the negative (representing the absence of an Illegal Landfill).

However, even after significant results in the prediction capabilities of the network, it is still unclear how this remote sensing scene classification task is performed. In order to understand what leads to the resulting predictions, a widely used strategy relies on **Class Activation Maps (CAMs [69])**, where the aim is to provide a visual representation of the parts of the input data that are thought to be more relevant by the trained network for the final prediction. Other than interpretability, applying this technique to the landfills detection task could help to assess the extension of the illegal landfills leading to better evaluation of the risk factor associated with the signaled sites. Furthermore, computing the CAMs of a classifier trained on a data set with only whole-image labels could be a first step leading towards the segmentation task automation, which can in turn be a stepping stone to weakly supervised object localization. This is particularly relevant considering again that the process of the creation of segmentation masks and bounding boxes around the significant elements of a scene is really costly and time-intensive, and up until now relies on human manual annotation of the datasets. Hence, after the training of the models, a subset of the original dataset composed of 596 images has been annotated with a total of 3411 segmentation masks surrounding various classes of specific waste objects. We then used this dataset to compare the annotated Ground Truths (GTs) with the various Class Activation Maps generated by the trained models. In particular, we used the Intersection over Union (IoU) metrics to evaluate the agreements between the areas highlighted by the CAMs and the GTs, the percentage of GTs covered by the CAMs, and the percentage of the CAMs that do not cover relevant wastes. Since the CAMs are produced fixing a threshold value constrained in $[0, 1]$, we repeated this analysis for each threshold in the range of the possible values with a step difference of 0.05.

The final results are that among the attention mechanisms ECA-Net was the one leading to higher improvements, with a $F1$ metric of 0.877 and an Accuracy of 0.923. Moreover considering the capability of highlighting the correct objects and areas with the CAMs, ECA-Net again performed better in every metric and considering every possible threshold.

Chapter 2

Technical Background

The following Chapter will first introduce the Computer Science's fields of Computer Vision and Machine Learning. Then it will be focused on a brief introduction of Neural Networks and Deep Neural Networks, explaining in more detail the architecture and the components of Convolutional Neural Networks, one of the most commonly used architectures to analyze visual data.

2.1 Machine Learning and Computer Vision

Over the past century the amount and the quality of visual data that we have been able to collect increased exponentially. We now have huge amounts of really complex systems that can capture images and/or videos, directly in digital form. But the main shift that we noticed is that, other than collecting the data, a much more difficult problem is to understand what's inside visual data.

Machine Learning is a study area of Computer Science, under Artificial Intelligence, that aims to provide algorithms able to improve automatically through experience and by the use of data. A more formal definition could be found in [42] and is:

> *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."*

Hence these programs do not need to be explicitly programmed to tackle a class of tasks but are instead able to automatically extract the relevant information embedded in the data, to then perform induction on new data. Machine Learning algorithms are especially useful to tackle those problems where it is still difficult or unfeasible due to computational complexity to develop conventional algorithms. Many interesting algorithms have proven to

be successful to solve a wide variety of tasks, but we are mostly interested in a class of algorithms that gained a lot of momentum and approval between researchers and engineers fairly recently, the Deep Neural Networks. Neural Networks are a class of algorithms that process the data with a logical structure similar to the human brain, and Deep Learning refers to the layering of many of those neural layers to obtain complex but also really powerful algorithms. This approach obtained remarkable results, even surpassing human expert performance, in many fields like speech recognition, natural language processing, drug design, medical image analysis, and computer vision.

Computer Vision is the interdisciplinary scientific field that investigates how machines can gain high-level understanding from digital images or videos. It touches fields like Physics, Biology, and Psychology, but, for the scope of this thesis, we're mainly interested in the perspective of Engineering, Mathematics, and Computer Science, with the aim to understand and automate tasks that the human visual system can do. From the '60s to the '90s various studies have been published with the aim to develop good techniques that could tackle the object recognition task. Those studies however remained a set of examples since they were not able to deliver functioning tools that were relevant for real-world applications.

From the start of the 21st century, the approach changed drastically. Following the improvements on the digital image capturing devices, and the explosion of the Internet, thus the availability of those digital data, researchers focused their efforts on a very important building block for object recognition tasks: the building of a set of benchmarks datasets. Those were the PASCAL and ImageNet datasets. With them, we were able to measure the improvements made over time on the object recognition problem. Another really important shift was the arisen interest and studies in the computer vision area that used machine learning techniques.

Starting from 1999, techniques in this field like Support Vector Machines, Boosting, and the first wave of **Neural Networks** started to gain momentum. In particular, a work that made a lot of contribution was [57], where Adaptive Boosting was used to make real-time face detections. From there, after only 5 years, cameras industries began to integrate this feature into their products. Many other researchers began to study how to find relevant feature spaces inside the images to tackle various problems such as object matching with Machine Learning models. In 2010 the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was started, where the best algorithms for object detection and image classification competed against each other to evaluate the progress in the field. In 2012 a huge breakthrough has been made in this competition, with the winner being AlexNet [31], a **Convolutional Neural Network** that spiked the interest in the use of Deep Neural Networks inside the Computer Vision field. Nowadays the most commonly used approach to tackle Compute Vision problems is by using Deep Learning Techniques.

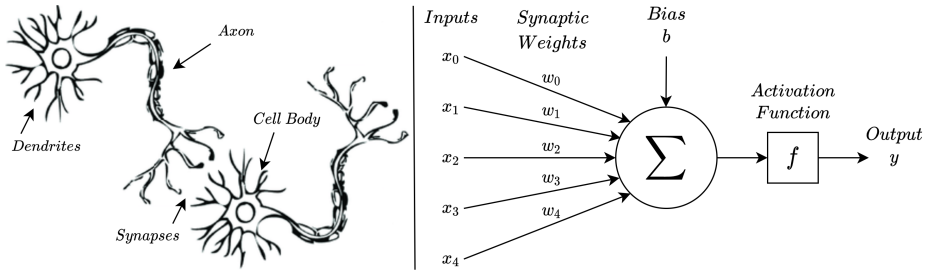


Figure 2.1: Example Neuron with 5 inputs $[x_0, x_4]$*, 5 weights* $[w_0, w_4]$*, the bias* b*, the non-linear Activation Function, and the output* y*.*

2.2 Neural Networks and Deep Learning

Neural Networks (NNs) are a class of models highly adaptable to solve various classes of problems, that is vaguely inspired by the human brain. They are formed by a collection of connected units, called neurons. These neurons resemble the neurons in the biological brain and were first introduced around 1957, inside the **perceptron** algorithm [52].

Figure 2.1 shows the comparison between the structure of the neurons and the structure of the perceptron, where each input could be seen as an axon coming from other neurons, the weights that are multiplied to those inputs as the synapses, the dendrites with the cell body as the function applied by the perceptron to its inputs, and the output as the outgoing axon of the neuron. The scope of the perceptron is to combine the input data with a set of coefficients (weights) that signals which inputs are more relevant, and that are learned during the training process. The activation function acts as a filter to determine if and with what magnitude should the signal pass to the rest of the network. More formally the overall function of the perceptron is the following:

$$y(x) = f\left(b + \sum_{i=0}^{N} w_i x_i\right) = f(b + w^T x) \tag{2.1}$$

where w_i is the weight for the i^{th} component of the input vector, b is a constant added term called bias, and $f()$ is the non-linear activation function. The activation function is really important since it enables the perceptron to learn a non-linear function of its inputs. These neurons are then connected to each other in order to develop a Neural Network that is capable of learning complex non-linear functions. The connections form organized collections that are hierarchically ordered, called layers. The connected architecture showed in figure 2.2 is a general example, where the signals travel from the first layer, called the **input layer** and composed by a number of

neurons equal to the number of inputs, to the last layer, called the **output layer** and composed by a number of neurons equal to the number of outputs, traversing a set of layers called the **hidden layers**. Usually, every neuron in a layer is connected to every neuron in the next layer and is completely independent from each other neuron of the same layer, thus no connection exists between neurons of the same layer. These kinds of layers are called **fully connected layers**.

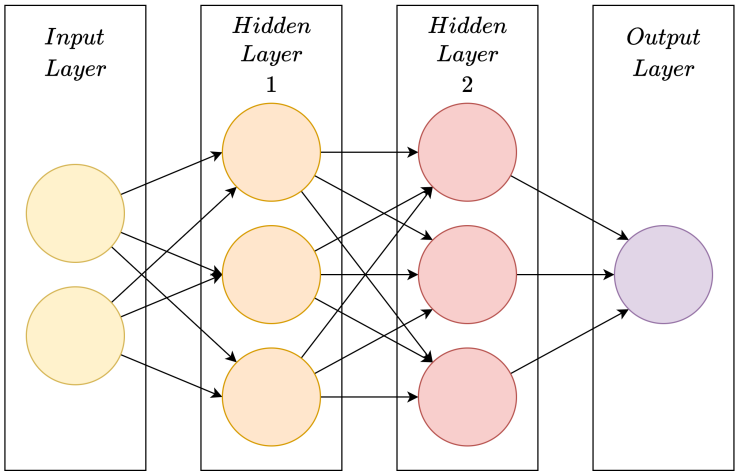


Figure 2.2: Example Layered Architecture of a Neural Network, with 2 Hidden Layers.

Given this structure of a Neural Network, in order to produce the desired output given an input sample, its weights and biases need to be adjusted accordingly. This process of learning the correct weights and biases for a given task is named the training process. It begins by making training and validation partitions of the given data, the first one used only to optimize the parameters and the second to tune the learning, evaluating the prediction error that the network is making with a **loss function**. This function is used to measure if the network is behaving well or not, looking at how close the behaviour of the network is compared to the desired one. Then, to obtain the optimal value for the parameters, the loss function is maximized/minimized, and the optimal value of the weights should be found by computing the gradient of the loss function with respect to the weights of the model and posing it equal to zero. But because of the network nonlinearity, the loss function is almost always non-convex, making a closed-form solution for the optimization problem not possible. Hence the common procedure is to use the technique of **backpropagation**, an algorithm based on the iterative method of **gradient descent** to update the values of the weights. The whole training of the network can therefore be summarized in a two-phase iterative procedure:

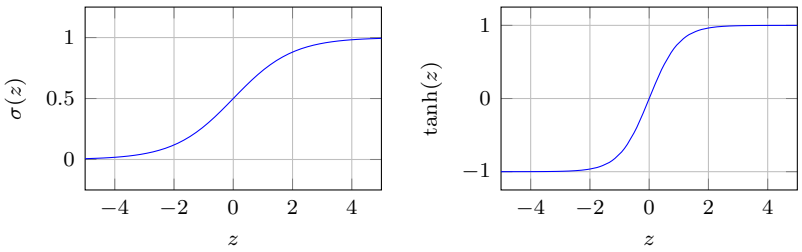
1. the **forward pass**, where the output of the network given an input is computed, saving the outputs of each node and computing the resulting loss of the output.

2. the **backpropagation**, where the gradients of the loss function are computed and the weights and biases of the neurons are updated, starting from the last layer to the first one, following the update rule given by the gradient descent algorithm.

Deep Learning methods are the ones that generally aim to enhance the networks' capabilities by using many layers progressively, to create far more complex and potent networks. While it was known for a long time that larger networks could bring performance improvement over simpler ones, Deep Neural Networks spiked in interest fairly recently, around 2012. Until then, the main problems that hindered their use were that, to contain the overfitting problem that naturally comes using more complex models, a huge amount of training data were needed. Moreover, these kinds of models are computationally intensive to train. The data availability problem was partially solved thanks to the modern solutions for data sharing and all the efforts made by the research community in order to build reliable benchmarks data sets. The computational complexity problems were instead tackled via the development of implementations of NNs that could be trained really fast thanks to the parallelization capabilities of modern high-performance GPUs.

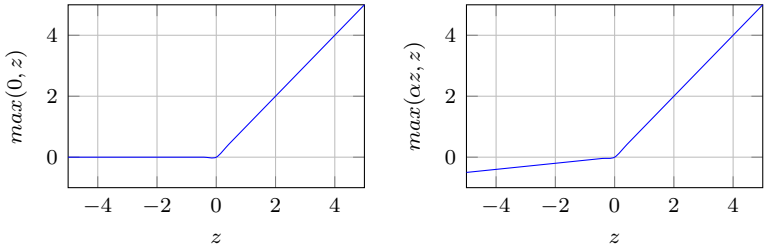
2.2.1 Activation Functions

As already stated, the activation functions are necessary in Neural Networks models, because without them the entire network is essentially just a linear regression model. Introducing these non-linear functions enables instead to learn complex non-linear patterns present in the data. Figure 2.3 shows some examples of the most used functions. The design choice of which ones to use could determine how well the network learns the training dataset, or, in the case of the output layer, defines the type of predictions that the model can make (e.g. step vs. logistic sigmoid).



(a) Logistic sigmoid.

(b) Hyperbolic tangent.



(c) Rectified Linear Unit.

(d) Leaky Rectified Linear Unit.

Figure 2.3: Commonly used activation functions.

2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of Deep Neural Networks that works particularly well with data that present a grid-like structure, thanks to the use of convolutional filters. The aim of a CNN is to progressively discover features and to combine them, from low-level features to a higher-level representation. In the following paragraphs, the building blocks that usually compose a CNN are described.

2.3.1 Convolutional Layer

The Convolutional Layer is the core layer of a CNN and is where the convolution operation is performed. An intuitive definition is that the convolution is a linear operator between two functions that express how the shape of one is modified by the other.

This operator is particularly useful in image analysis. A digital image is usually represented by a 3-dimensional matrix of pixels of size $H \times W \times C$, where H represents the height of the image, W the width, and C the colour channel (e.g. usually $C = 3$, that are the RGB channels). Hence an image could also be seen as a 2-dimensional function, taking in input a row and a column and returning the channel value of the pixel in that position. With this said, a convolution could be seen as the operation $f * g$, where the first function f is the image, and the second function g is a filter, for-

mally known as kernel. Depending on the size and values of the filter we can obtain a wide variety of effects, for example blurring the image, by means of averaging each pixel to the value of its neighbours, or edge detection, where the filter identifies the pixels where image brightness is discontinuous. In general convolving an image with the right kernel can be seen as a method to extract meaningful features from the picture. This is because the convolution operation is a means to highlight local relationships in a structured set of data, and images in particular have a clear 2-dimensional structure, where pixels that are close to each other are strongly correlated. In practice, each kernel has a fixed spatial extension that is defined by the kernel size and a depth equal to the input depth. The spatial extension is usually small and contained with respect to the extension of the original image. With the convolution operation the filter is slid across the input width and height to cover all the input volume and the dot product between the entries of the filter and the input at each position is computed. This produces a 2-dimensional activation map that represents the responses of that filter at each spatial position. Two other hyperparameters P and S, where P is the padding and S is the stride, define the output dimension of a convolution operation. The stride represents the step used when sliding the filter through the input. The padding instead is a parameter that defines the amount of zero-padding applied to the input and allows us to control the input spatial size, to obtain an output that preserves the same width and height of the original input. With W equals to the input width, K equal to the kernel size, we have that the spatial size of the output is:

$$W_{output} = \frac{W - K + 2P}{S} + 1 \tag{2.2}$$

In each convolutional layer more than one kernel can be employed simultaneously (conventionally the number of kernels is equal to a power of 2). The numbers of different filters used in each layer represent another hyperparameter, the depth of the convolutional layer. From each of these filters a different feature is learned, and the concatenation of the activation maps produced by every filter is called the feature map.

The main benefit of this kind of architecture compared to a fully connected layer is that it encode in its structure the knowledge of having inputs with a clear spatial structure and permits to greatly reduce the number of parameters to be learned. This is due to the fact that each filter learns to detect a feature with a number of parameters defined by the kernel size, and then it can be applied to the whole image.

Non-linearities (the activation function) are again applied after each convolution since the convolution is linear and the Network would like to learn a non-linear function.

2.3.2 Pooling Layer

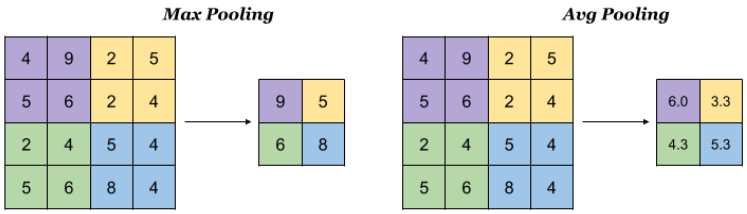


Figure 2.4: Examples of pooling layers with window sizes and strides equals to 2. On the left Max Pooling, on the Right Average Pooling.

Another kind of layer that is commonly placed right after a convolutional one is the pooling layer. Its aim is to reduce the spatial size of the input by subsampling it, resulting in a more concise representation of the data. It does so by computing a statistic over the elements inside a window of fixed size that is moved across the spatial dimension, leaving the depth unchanged. The purpose is to improve noise robustness and increase the size of the receptive field (*i.e.* area of the input covered by the filter) in deeper layers without explicitly increasing the size of the filter. This lead to models that can promote the depth of the network instead of promoting the overall width, that usually provides benefits in term of performances. The most common types of pooling are max-pooling and avg-pooling, and their functioning is illustrated in figure 2.4.

A particular type of layer that derives from these pooling layers is the global ones. They are the versions in which the pooling length, or size of the pooling windows, is equal to the size of the input, resulting in an output that is a 1-dimensional vector. The most used one is the Global Average Pooling, that has numerous application that will be introduced afterward.

2.3.3 Fully Connected Layer

The last layer is the one that is tasked with learning the correct prediction to associate with the extracted features. This is usually done with a fully connected layer like the ones already explained previously. In order to use it, the last feature map is first flattened, to obtain a 1-dimensional vector from the 3-dimensional one. Then a fully connected layer is used to obtain a vector with length equal to the number of object classes that the model would like to recognize, where the i-th values represent the probability that the i-th object is present inside the input image.

2.3.4 Regularization Techniques

Data augmentation

It is well known and also a theoretical result of Machine Learning that, in order to contain the variance (i.e. over-fitting), one could simply increase the number of samples. However, enlarging the data set is not always possible and most of the time requires a costly process. The data augmentation approach tries to emulate the addition of new samples by transforming the ones already available. The transformation usually used are horizontal and vertical flips, rotations of various degrees, zooming, change in brightness, and warping angles. The result is an augmented data set that can be used to train models less prone to over-fitting.

Batch Normalization

The idea behind Batch normalization (batch norm [25]) is to normalize the inputs of each layer to obtain a mean output activation of zero and a standard deviation of one, exactly like how the input of the networks are standardized. Hence this batch normalization layer is usually placed between the convolution and the activation. The benefits of this technique are many and among them, the main ones are faster training thanks to quicker convergence, the possibility to use higher learning rates, less sensitivity to the starting weights initialization and making more activation functions viable since it regulates their input values.

2.3.5 Transfer Learning

> *"Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned."* [56]

The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task. It is especially useful given that there are usually high computational, time and data availability requirements to train a model from scratch. Hence Transfer Learning intervenes to start from an initial performance on the target task that is higher than the one of an ignorant agent. Moreover, it decreases the amount of time needed to train the learner on the target task and also helps to achieve higher performances compared to the trained learner without transfer. Figure 2.5 illustrates the improvements that could be achieved with transfer learning. It is important to understand that, in order to reach the desired benefits with this technique applied to deep learning settings, the features that are learned from the source task need to be general, meaning suitable for both the tasks instead of being specific to the source task.

The Transfer Learning technique is commonly used in the Computer Vision field and especially in image classification tasks. In fact, usually, researchers that develop new models, train them on really large and challenging datasets such as ImageNet [14] and then release their final model under a permissive license for reuse. Given that those models were trained on a really large corpus of images and learned to make predictions on a wide variety of classes, it is reasonable to think that they learned to extract general enough features, especially in earlier layers of a CNN. From this, future research can then download, incorporate and adapt these pre-trained models and apply them to different tasks.

Another technique that could be used in conjunction with transfer learning is freezing some amount of initial layers of the network. This means that, while using a pre-trained model, we halt the weights update for some of the initial layers, training only the later layers that represent the high-level features of the input images. This once again is motivated by the fact that low-level features should already be learned by the pre-trained model on a wider dataset and should be general enough, meaning that there should be no need to refine these features on our available data.

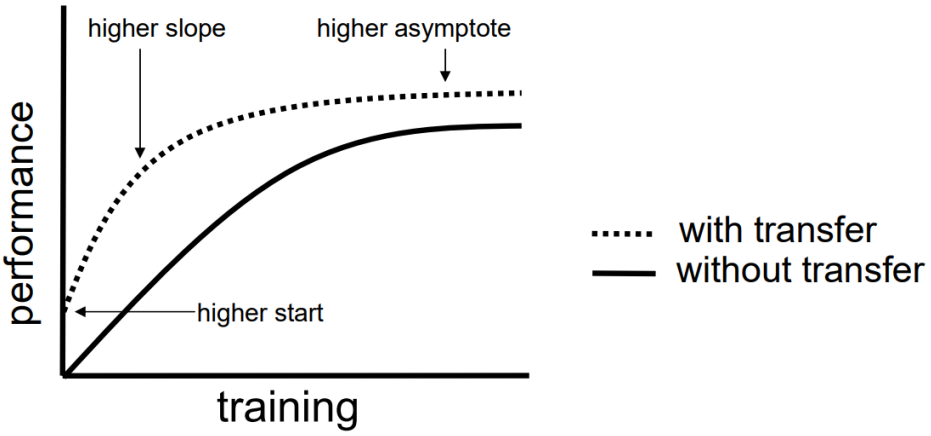


Figure 2.5: Graph that shows the three main benefits that could be achieved with transfer learning [56]

Chapter 3

Related Works

In this Chapter, in Section 3.1 we first give a brief introduction and historical background of some of the research that has been conducted with the aim of detecting illegal landfills to prevent the occurrence of environmental hazards. Then, in the following sections, we explain the specific techniques and CNN architectures that are used in this thesis, presenting the main key points of the original papers.

3.1 Illegal Landfills Detection

Starting in the '70 and '80, when there was still a restricted amount of data available, studies like [17] and [15] were conducted in the U.S. and proved that remote sensing imagery could contribute to the temporal assessment of landfills existence and cataloguing landfills extension. A problem that would become more relevant in later years was that the classification of the aerial images was a costly and time-consuming process, that relied entirely on human experts. In fact, going forward in the years, the availability of such data increased exponentially both in quantity and in quality of the data, while the computed-aided identification of landfills area is still an ongoing study field. Starting around '00 the first automated techniques for landfills identification started to being developed. In [54] the authors conducted a study in the Italian region of Veneto, where they used a stress vegetation index computed from multispectral satellite images taken by *IKONOS*[1] to define candidate sites that could host waste material. Then they combined the candidates identified through remote sensing with GIS information (e.g. street network, population density) and historical aerial photographs to select the most likely contaminated sites. Their approach proved that vegetation stress is a relevant indicator of landfills' presence and that the integration of GIS with remote sensing data is a key instrument to discover possible illegal dumping sites. Subsequent research [4] on the same territory

[1] Commercial high-resolution imaging satellite, provides 1m spatial resolution imagery.

used GIS information like road networks, low population density, and presence of industrial sites, in combination with a dataset of known landfills, to perform a spatial statistical analysis to select factors and criteria that could identify a waste deposit. With it, they developed a map describing the likelihood of occurrence of an illegal landfill and then produced a set of candidate illegal landfills sites by combining the above-mentioned map together with remote sensing technique previously developed [54] to further narrow down the possible illegal landfills locations. Their final results were that *84%* of the illegal landfills of the validation set were located in the high-probability area, while only *5%* of the validation illegal landfills were located in the low-probability area. Moreover, they were able to identify a total of 1199 candidate illegal landfills, 738 of them contained within the high-probability area. Information coming from GIS has been used also in other studies [29][39] showing that many factors, the most influencing being demographic and spatial characteristics of the territory, are correlated to the presence of illegal landfills.

In more recent years, in the Computer Vision field, Deep Learning methods have been applied to the waste classification and waste dump detection problem. Most of the literature focuses on recognizing the waste types to help with the separation of recyclable waste or recognizing wastes and rubbish from street-level images.

Researches like [1], [48], and [40] focus their efforts on aiding the waste management process during the step that involves the separation of different waste components. To do so, they built classifiers able to differentiate the waste contents of an image between various classes like plastic, paper, and metal, obtaining high classification accuracies. Other works focused on some particular type of wastes. A first example could be [45] where the authors built a CNN, with a region-based CNN (R-CNN) on top, to classify and identify the sizes of e-waste equipment inside pictures. Another example is [28], where the authors examined lakes and rivers in Bosnia and Herzegovina using Unmanned Aerial Vehicles (UAVs) to collect images, which were then used to develop a deep learning algorithm for automatic floating plastic extraction from those orthophotos. The resulting models provided reliable estimates of the area and volume of the plastics, and were also able to provide high accuracy in the classification of the different kinds of plastics (like Oriented Polystyrene (OPS), Nylon, Polyethylene terephthalate (PET), and general plastic).

Looking at the works on waste detection from street-level images, in a research of 2016 [43] the authors created SpotGarbage, a smartphone application capable of detecting scattered waste present in street-level images. To develop it they used a CNN named GarbNet, based on AlexNet [31],

that was trained on the Garbage in Image (GINI) dataset. This dataset has been created ad-hoc for this project and contained 450 images used for binary classification of street-level garbage. Training GarbNet on this dataset, starting from a pretrain of AlexNet, they obtained 87.69% mean accuracy. Moreover, using a patch-based classification approach, they were able to approximately highlight the regions in the image that correspond to garbage, generating other than the prediction for the whole image also a polygonal bounding box containing the wastes. Lastly, they optimized the trained model reducing the memory usage and the prediction times, in order to create a lightweight application that could be run directly on smartphones. A later work [59] of 2018 used data fusion and data augmentation to create a dataset of 816 urban scenes, that have been used to discriminate between garbage and background classes such as pedestrians, vehicles, buildings, roads, and lawn. Then a Faster R-CNN [18] with a ResNet [20] backbone, pretrained on the COCO [37] dataset, was used to perform waste detection. With this architecture they were able to obtain a precision of 89% on their dataset. Really recently in 2020 another research [13] focused on garbage detection in video streams. In it, the authors created a dataset of 3974 images, with 5535 annotated bounding boxes, for the four classes of garbage bag, garbage dumpster, garbage bin, and conglomerate of waste objects. Then, using an improved version of the You only look once (YOLO [49]) network, the YOLOv3 [50], they trained a model that was able to obtain a mean average precision of around 60% on the validation set. Another research around the same years [65] aimed to build a real-time monitoring system, focusing on surveillance camera videos to recognize the garbage dumping action made by a pedestrian. In order to do it the authors combined pedestrian detection, the tracking of carrying objects, and a voting mechanism to determine if the dumping action has been performed. Their evaluations have been performed on a dataset obtained from the received videos of the local government closed-circuit television (CCTV) cameras, and obtained an F1 metric of 0.71, corresponding to a precision of 0.79 and a recall of 0.64.

While most of the literature on garbage detection using Deep Learning methods used data coming from street-level images or videos, another interesting approach is applying these methods for the recognition of this kind of environmental hazards from remote sensing aerial images. From a technical standpoint, this is motivated by the promising results that those methods have shown, and once again is driven by the wider availability of high definition satellite imagery and the advances that have been made in the general application of deep learning methods to remote sensing data. A recent survey [9] shows the rapid evolution of the various methods that have been developed to tackle the task of Remote Sensing scene classification, which aims at labelling remote sensing images with a set of semantic cate-

gories based on their contents. In particular, it shows the challenges of this task and explores the methods based on CCNs developed to tackle it, giving also a brief introduction to the various benchmarks datasets that have been developed to measure the improvements on this kind of task. As stated before, benchmark datasets are an important building block that drives innovation, and starting from 2010 thirteen new datasets for remote sensing image scene classification have been made publicly available. Among them, the first one was the UC Merced [64], that have been widely used in many following studies of the field. Another really relevant one published in 2017 is the AID dataset [62], where images belonging to 30 different classes have been sampled from different regions and during different seasons, to increase the intra-class diversities of the data. The task of expanding the available benchmark datasets is still ongoing, with a parallel rising interest towards the application of unsupervised models, due to the laborious task of image annotation that is still entrusted to human experts.

Building models from satellite data is particularly important to tackle the problem of illegal landfills proactive detection, since it would enable to then run trained models periodically on newly collected data and perform mass-scale territory monitoring campaigns. In this direction, our research focused on the scene classification of a remote sensing image, to recognize if it contained an illegal landfills or not. Moreover, the novel element is that we experiment with attention methods, that have been already applied to Remote Sensing scene classification tasks (e.g. [66], [68], [55]) with great results, but not in this particular case of waste and IL detection that pose novel challenging problems.

3.2 Residual Neural Networks

As stated in Section 2.2, increasing the network depth should increase its capability to extract complex and meaningful features and should lead to an increase of the model performances. But this approach, even with the computational capabilities of modern hardware, has not been trivial. Deep Networks with many layers tend to be really hard to train, mainly because of the vanishing gradient problem. The issue is that, as the gradient backpropagates to earlier layers from the loss function, it exponentially decreases, preventing the weights update and thus hindering the learning process. These are the reasons why the publication of deep Residual Networks (ResNet [20]) is widely considered as another milestone work of the last few years like AlexNet [31]. The core of ResNet is using an identity shortcut connection, that aims to ease the training process of really deep networks. The shortcut connection and the architectural block that it creates, the residual block, are illustrated in Figure 3.1. The residual block can be more formally

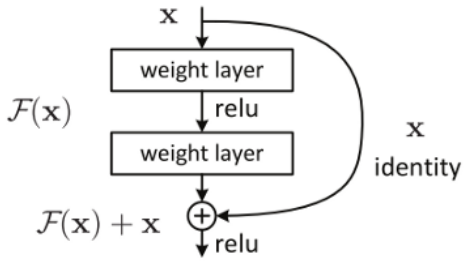


Figure 3.1: Illustration of the skip connection and residual block concept in Residual Networks [20]

defined as:

$$y = \mathcal{F}(x, \{W_i\}) + x \tag{3.1}$$

where x is the input vector, y the output, and $\mathcal{F}(x, \{W_i\})$ represent the mapping to be learned. In Figure 3.1 the actual function is $\mathcal{F} = W_2\sigma(W_1x)$, where σ is the activation function ReLU [44] (for the sake of simplicity biases are omitted). Then the skip connection, element-wise addition, and an additional ReLU are performed to obtain the operation $y = (\mathcal{F}+x)$, where x and $\mathcal{F}$ are considered to have the same dimension (different dimensions could be handled with padding or other methods). The ladder operation does not involve any additional parameter nor add any significant increase in computation complexity, with respect to the equivalent model without the skip connection. This technique is also applicable both to fully connected layers and to convolutional layers, and it is quite flexible in the number of layers that can be placed inside the residual block. The purpose of this operation can be interpreted intuitively as a means to let the gradient flow more easily and faster during the backward path, compared to the same network without the skip connection. As already stated this method adds no burden to the computations and memory requirements, leading to an architecture that is able to empirically perform better on training error as we add more layers, as theoretically and intuitively expected.

The authors proved empirically their results by constructing a set of differently sized ResNet showed in Figure 3.2, and even explored a model with more than 10^3 layers, resulting in no optimization difficulties during the training phase. Considering that in comparison AlexNet [31] had in total 8 layers, the ResNet approach went really far.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 3.2: The table describes all the architectural components and Floating Points Operations (FLOPs, measure of operations done in the network) of the ResNet architectures with 18,34,50,101, and 152 layers [20]

3.3 Attention Methods

For the scope of this thesis, we investigated and implemented some techniques that in recent years have made profitable results applied to CNNs, the attentions methods. They have been developed once again by looking at the human learning patterns and how the human visual perception is characterized by a selective process in which the most relevant elements are identified to capture the overall structure of the scene [51][12]. **Attention mechanisms** are the techniques that try to simulate the way in which humans understand and perceive images. The aim of these methods is to concentrate the allocation of the available computational resources on the smallest and the most informative parts of the input data [32][26], to carefully and quickly discover the most important features. This approach has been applied with great results to many tasks, like in large-scale classification [23][58][60][6], sequence learning [5][41], object localization [6], image understanding [27], and captioning [63][8].
In the following Sections we present the specific implementation of the attention modules that have been studied and applied for this research.

3.3.1 Squeeze-and-Excitation Networks

The first mechanism that we studied is the Squeeze-and-Excitation Network (SE-Net [23]) architecture, which consists of an attention module that could be placed on top of any pre-existing CNN backbone (Figure 3.3). Its aim is to exploit the inter-channel relationships to learn which channels are more (or less) important inside the feature map that receives as an input. To do so, the first step, named *squeeze*, is to aggregate the spatial information

of the input into a channel descriptor and is performed by global average pooling (GAP). Then the next step is the *excitation* one, where a gating mechanism captures the dependencies between the channels that are present in the descriptor. The implementation of this gating mechanism is made by a Multi-Layer Perceptron (MLP), which consists of two fully connected (FC) layers with a ReLu [44] non-linearity between them. The authors have chosen to use this specific implementation in order to contain the number of additional parameters to $2 \times \frac{C^2}{r}$, where r is the reduction ratio, obtained with a dimensionality-reduction layer followed by the ReLU and then a dimensionality-increasing layer returning to the original channel dimension. The overall scope of this mechanism is to make the CNNs able to learn non-linear interactions that occur between the various channels. After the *squeeze* and *excitation* operations, followed by a sigmoid, the output is a vector that states which channels are more relevant in the original feature map. This vector is hence multiplied with the original input of the whole module, using element-wise multiplication, to obtain an output feature map where the most relevant channels are emphasized.

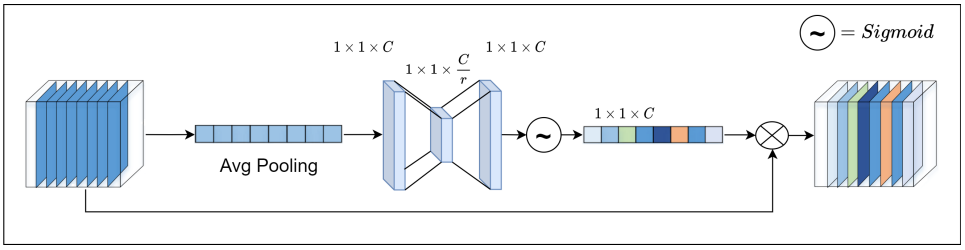


Figure 3.3: Squeeze-and-Excitation channel module.

More formally we have that, given an input feature map $\boldsymbol{F} \in \mathbb{R}^{C \times H \times W}$ the module produces a 1D channel attention map $\boldsymbol{M_c} \in \mathbb{R}^{C \times 1 \times 1}$ and the output of the model $\boldsymbol{F'}$ can be summarized as:

$$F' = M_c(F) \otimes F \tag{3.2}$$

where $\otimes$ denotes element-wise multiplication. The channel attention Map $\boldsymbol{M_c}(F)$ is instead computed with the following formula:

$$\boldsymbol{M_c}(\boldsymbol{F}) = \sigma(MLP(AvgPool(\boldsymbol{F}))) = \sigma(\boldsymbol{W_1}(ReLU(\boldsymbol{W_0}(\boldsymbol{F^c_{avg}})))) \tag{3.3}$$

where $\boldsymbol{\sigma}$ denotes the sigmoid function, $\boldsymbol{W_0} \in \mathbb{R}^{C/r \times C}$ and $\boldsymbol{W_1} \in \mathbb{R}^{C \times C/r}$ are the MLP weights, $\boldsymbol{ReLU}$ is the Rectified Linear Unit inside the MLP, and $\boldsymbol{F^c_{avg}}$ is the spatial context descriptor obtained with the average pooling. From here it can be easily seen the effect of setting the hidden activation size to $\mathbb{R}^{C/r \times 1 \times 1}$, that is to reduce the parameter overhead with a reduction ratio $\boldsymbol{r}$.

In the evaluation of this module on the task of image classification on the ImageNet [14] dataset they showed that ResNet50+SE was able to reduce the top-5 validation error by 0.86%, and the top-1 by 1.51%, compared to the same standalone backbone. Experimenting also with its addition to ResNet101, this attention enhanced architecture was able to outperform even the deeper ResNet152, by 0.27% in top-5 error and 0.04% in the top-1.

At last, is important to notice that this research was among the first to successfully implement attention methods for CNNs, and most subsequent iterations, even the most complex models, are often inspired by the *squeeze* and *excitement* mechanisms originally proposed in this paper.

3.3.2 Efficient Channel Attention

The second attention mechanism that we studied is Efficient Channel Attention module (ECA-Net [58]), where the authors focused their effort on lowering the computational complexity of the module instead of proposing a more sophisticated one, trying to overcome the performance and complexity trade-off. In order to do so, they looked at the SE-Net module [23] and proposed to change the gating mechanism, that was a Multi-Layer Perceptron, into a 1D convolution. In fact, the parameters of a 1D convolution are way less than that of an MLP, and they were also able to improve the performances, mainly thanks to a better exploitation of local cross-channel interaction.

To develop this module the authors first noticed that dimensionality reduction hinders the ability to learn effectively the channels relationships. To prove that is better to avoid this stratagem they looked at variations of the SE-Net where the MLP had no reduction ratio and noticed an increase in the performances. Moreover, they noticed that, among the variations without dimensionality reduction, using a single FC layer was the best performing mechanism thanks to its exploitation of the cross-channel interactions. This mechanism however increased the number of parameters by a large amount, leading to C^2 parameters. Hence, in order to capture the cross-channel interaction but limit at the same time the number of parameters, they started investigating directly the channel features. There they found a pattern, that is a local periodicity inside the channel features. Therefore they then focused on capturing *local*-cross channel interactions, which consist of considering only the interaction between a channel and its nearby neighbours, effectively reducing the number of parameters needed. This leads to an attention module that considers the interaction between each channel and its k neighbours, which involves $k*C$ parameters. To obtain an even less complex and more efficient module the authors then decided to let all the channels share the same learning parameters, leading to an architecture that could be implemented by a fast 1D convolution with kernel size equal to k. The overall complexity of this module is greatly reduced, with only k parameters

needed. An example of this module can be seen in Figure 3.4.

The kernel size k of the convolution is a crucial parameter since it determines the coverage of the local cross-channel interaction. Moreover, this parameter may vary against convolution blocks with different channel numbers and various CNN architecture. Therefore the authors also proposed a function of the channel dimension that is able to adaptively select the kernel size that should be used, given a chosen CNN backbone and the input channel size of the convolution block.

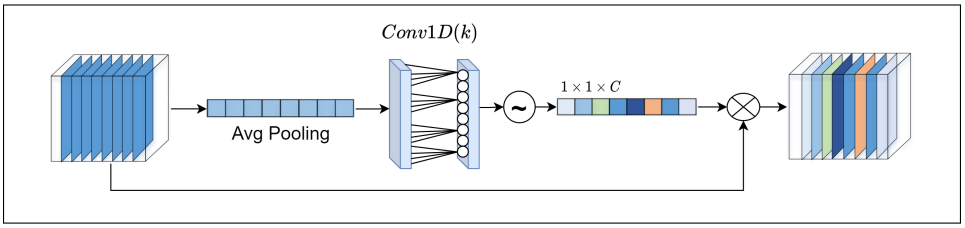


Figure 3.4: Efficient Channel Attention channel module.

Their evaluation in the image classification task has been made on the ImageNet-1K dataset [14]. The most relevant results are that ResNet50+ECA surpassed ResNet50 by 2.28% in terms of top-1 accuracy, while also maintaining similar model complexity in terms of network parameters, floating point operations per second (FLOPS), training speed, and inference speed. Compared to SE-Net and CBAM [60] with ResNet50 as backbone, it surpassed their performance in top-1 accuracy, while maintaining a higher training and inference speed, and especially maintaining a number of parameters that is almost equal to the ones of the original backbone.

3.3.3 Convolutional Block Attention Module

The last attention mechanism studied in detail is Convolutional Block Attention Module (CBAM) [60]. Here the authors focused their efforts on two strategies, the first one being enhancing the feature aggregation of the channel-wise attention modules and the second one the development of another attention module that is responsible of trying to learn the spatial attention. Their first result was obtained by modifying the channel attention mechanism present in Squeeze-and-Excitation. They were able to verify that the use of max-pooled features together with the average-pooled ones in the squeeze operation was able to yield finer-grained channel-wise attention, because max-pooling was able to gather other salient clues about distinctive object features. The resulting formula is:

$$\begin{aligned} \boldsymbol{M_c}(\boldsymbol{F}) &= \sigma(MLP(AvgPool(\boldsymbol{F})) + MLP(MaxPool(\boldsymbol{F}))) \\ &= \sigma(\boldsymbol{W_1}(ReLU(\boldsymbol{W_0}(\boldsymbol{F^C_{avg}}))) + \boldsymbol{W_1}(ReLU(\boldsymbol{W_0}(\boldsymbol{F^C_{max}})))) \end{aligned} \tag{3.4}$$

where $\boldsymbol{M_c} \in \mathbb{R}^{C\times1\times1}$ is the channel attention map, $\boldsymbol{\sigma}$ denotes the sigmoid function, $\boldsymbol{W_0} \in \mathbb{R}^{C/r\times C}$ and $\boldsymbol{W_1} \in \mathbb{R}^{C\times C/r}$ are the MLP weights, and $\boldsymbol{ReLU}$ is the Rectified Linear Unit inside the MLP. $\boldsymbol{F^C_{avg}}$ and $\boldsymbol{F^C_{max}}$ are the spatial context descriptors obtained with the average pooling and max pooling respectively. As it can be seen by the formula, both descriptors are forwarded to the shared MLP to produce the channel attention map. Figure 3.5 shows the architecture of this channel module.

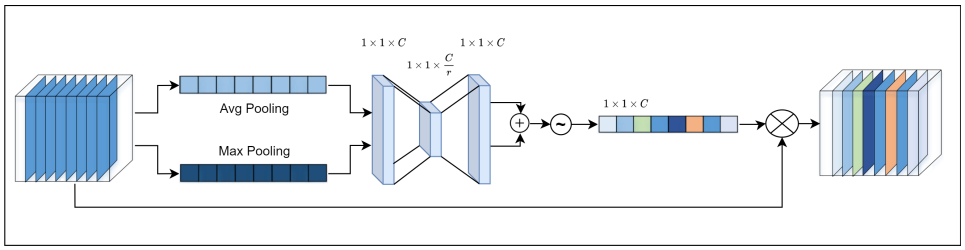


Figure 3.5: Convolutional Block Attention Module, channel module.

With CBAM the authors also introduced a spatial attention module, which is shown in Figure 3.6. This additional module's aim was to learn "where" to focus. In order to do it, it first performs a feature aggregation similar to the squeeze operation present in the channel module. This is done by applying average-pooling and max-pooling along the channel axis of the input feature map, and then concatenating the outputs to obtain a spatial feature descriptor. Then a convolutional layer followed by a sigmoid is used to learn which part of this feature descriptor, i.e. which part of the spatial region, is more informative. More formally we have that we generate a spatial attention map $\boldsymbol{M_s} \in \mathbb{R}^{H\times W}$ with the following formula:

$$\begin{aligned} \boldsymbol{M_s}(\boldsymbol{F}) &= \sigma(f^{7\times7}([AvgPool(\boldsymbol{F}); MaxPool(\boldsymbol{F})])) \\ &= \sigma(f^{7\times7}([\boldsymbol{F^S_{avg}}; \boldsymbol{F^S_{max}}])) \end{aligned} \tag{3.5}$$

where $\boldsymbol{F^S_{avg}}, \boldsymbol{F^S_{max}} \in \mathbb{R}^{1\times H\times W}$ denotes the two 2D maps of the average-pooled and max-pooled features across the channel, the σ denotes the sigmoid function, and $f^{7\times7}$ represents a convolution operation having kernel size of 7×7.

The generated spatial attention map is then applied to the original input feature map of the spatial module, via element-wise multiplication, to highlight in the output feature map the regions that are more important. To combine the channel attention module and the spatial one the authors tried the following configurations:

1. Sequence with channel module followed by the spatial module
2. Sequence with spatial module followed by the channel module

3. Parallelize the modules

Between these configurations, the best results were obtained by using the two modules in sequence, with channel first and spatial afterwards, resulting in the formula:

$$\begin{aligned} F' &= M_c(F) \otimes F \\ F'' &= M_s(F') \otimes F' \end{aligned} \tag{3.6}$$

where $\otimes$ denotes the element-wise multiplication.

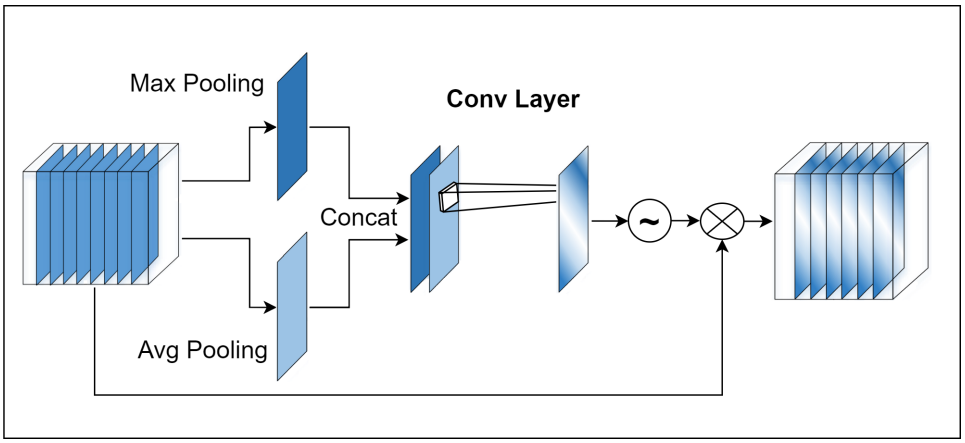


Figure 3.6: Convolutional Block Attention Module, spatial module.

In their evaluation the authors proved that this attention module is able to outperform both standalone ResNet50 and the SE enhanced version, in the image classification task over ImageNet-1k [14]. With a particular relevance for this thesis, another work [67] showed instead, among the others findings, that the addition of CBAM to ResNet50 is beneficial also in the tasks of scene classification done over the UC-Merced [64] and the AID [62] datasets, especially for the second examined dataset, that contains more complex scenes and more intra-class diversities.

3.4 Interpretability and Class Activation Maps

Most of the Deep Learning models have been constructed as black-boxes that do not expose their internal operations. In order to interpret their predictions many techniques have been developed. In particular Saliency Masks aim to solve this *outcome explanation problem*, by providing a visual representation of the parts of the input data that are thought to be more relevant by the trained network for the final prediction. Belonging to this class of techniques the most popular is Class Activation Mapping. The Class Activation Maps (CAMs [69]) of a given category are the discriminative regions

of the image that are used by the CNN to identify such category. They're especially useful to interpret what the CNN has learned and its ability to perform the classification task, again understanding what regions have the most impact on the prediction. Moreover applied to the landfills detection task, visualizing the relevant objects and their extension could help in the evaluation of the risk factor associated with the signaled sites. Furthermore, computing the CAMs of a classifier trained on a data set with only whole-image labels could be a first step leading towards the segmentation task automation, that can in turn be a stepping stone to weakly supervised object localization.

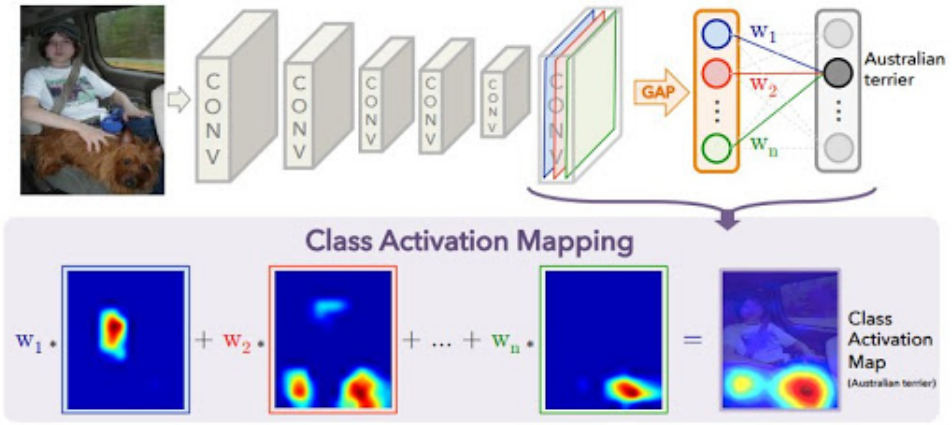


Figure 3.7: Class Activation Mapping. The predicted class score is mapped back to the last convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions. In this example, the image region containing the Australian terrier is highlighted

Class Activation Mapping is the technique used to generate the CAMs. First, it uses a Global Average Pooling (GAP) layer placed right after the last convolutional layer of the network. With this structure, the weights that we obtain with the fully connected layer right after the GAP can be projected back to the convolutional feature maps. Then a weighted sum between all the features map scaled by their respective weights leads us to the CAM matrix. This procedure is shown in Figure 3.7. More formally, given a point (x, y) and given $f_k(x, y)$ as the last convolutional layer activation of unit k, using GAP we obtain $F^k = \sum_{x,y} f_k(x, y)$. Then, to classify multiple classes, the weights w_k^c corresponding to class c for unit k are fed to a Softmax (to differentiate between the multiple classes). Then with S_c as the input of the

Softmax for class c and P_c as the output we have:

$$\begin{aligned} S_c &= \sum_k f_k(x, y) \\ P_c &= \frac{\exp(S_c)}{\sum_c \exp(S_c)} \end{aligned} \tag{3.7}$$

As shown by the equations, w_k^c can be seen as the importance of the feature map F_k for class c. Finally, we can define the CAM value in each spatial point (x, y) for class c as $M_c(x, y)$:

$$M_c(x, y) = \sum_k w_k^c f_k(x, y) \tag{3.8}$$

The spatial size of the output is given by the size of the last convolutional layer feature maps, that is smaller compared to the size of the original input image. Therefore the class activation map is upsampled with bilinear interpolation to match the original input size. Finally, this output is normalized by min-max scaling, to obtain values in the range $[0, 1]$.

Chapter 4

Datasets

For the scope of our project we started from an already present binary classification dataset, that has been constructed using the orthophotos generated by various satellites and given to us by the Italian agency *Agenzia per le erogazioni in agricoltura* (AGEA). These given source images cover a wide area of the north Italy provinces of Lodi, Brescia, and Pavia. From those orthophotos, a sub-sampling procedure generated the images that compose our dataset. The coordinates of known illegal landfills signaled by the Italian agency *Agenzia Regionale per la Protezione dell'Ambiente* (ARPA) Lombardia were used to crop the orthophotos and generate the positive samples, while the negatives ones are generated from random coordinates in the same analyzed provinces. The positive class represents the presence of an Illegal Landfill in the image, and the negative one the absence (mutually exclusive classes). A total of 2993 images have been generated, each one having a random scale between 600×600, 800×800 or 1000×1000 pixels. In the generated dataset, 990 areas correspond to positive samples, and 2003 to negative samples. Figure 4.1 shows example images of an Illegal landfill and a negative sample from the dataset.

Figure 4.1: Two examples of a positive and negative sample side by side.

The dataset is divided into 3 splits based on the geographical position: ∼75% training, ∼12.5% validation, and ∼12.5% testing.

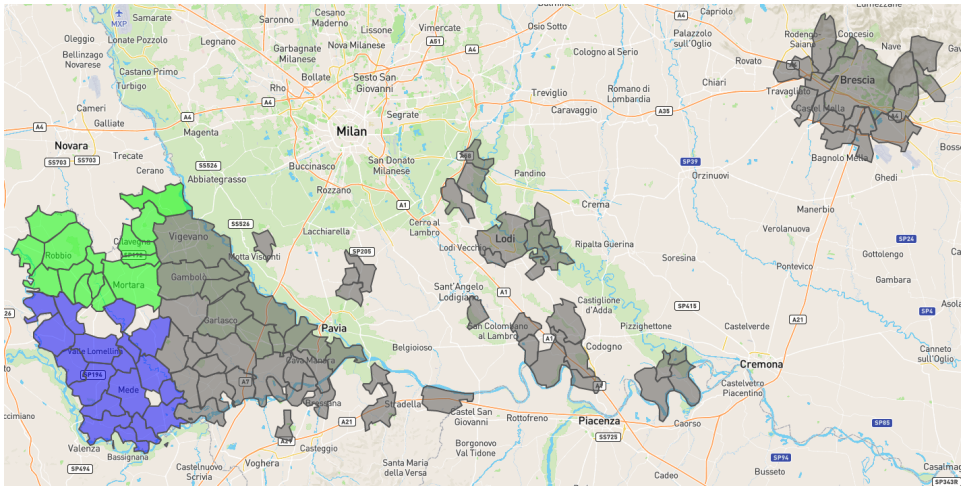


Figure 4.2: Map showing the Provinces from which the coordinates of the dataset have been selected. Grey ones are used for the Training split, Blue ones for the Validation and Green for Testing.

Then, starting from this dataset, a subset of 596 images has been annotated with polygonal segmentation masks framing the suspicious objects in each scene. The images that have been annotated are all positive samples of the original dataset and the Table 4.1 indicates how many images come from each split of the source dataset. All the positive samples of the Training dataset have been annotated. The choice of using polygonal segmentation masks instead of rectangular bounding boxes has been made in order to better capture the geometry of the ground truth wastes. Moreover, it is trivial to revert to the bounding box analysis from the polygonal segmentation if needed.

Annotation Dataset		
Training	Validation	Testing
398	92	106

Table 4.1: Sample of the annotated dataset provenience.

The classes of the objects captured by the annotations have been based on the knowledge that ARPA Lombardia shared, which provided insights on what are the common elements that could indicate the presence of an illegal landfill. The final choice was to use the following 11 categories:

- **scattered wastes**: wastes typically composed of scattered trash, rubble, and wreckage.

- **pallets**: flat transport structures typically made of wood, which are usually stockpiled on top of each other. It is easy to find in areas where wastes are amassed, usually scattered instead of being stored in columns.

- **intermediate bulk containers (IBC)**: used for containment of liquids, that could be a particularly hazardous waste due to the possible toxic nature of the liquid.

- **dumpsters**: containers that are opened on top, typically full of wastes, that can be commonly found in groups inside illegal landfill sites.

- **containers**: containers closed on top, that are commonly used for railway or naval transportation. Like dumpsters they can be found in groups inside illegal landfills.

- **tires**: abandoned tires made of plastic that take ages to be disposed of by the environment.

- **plastics bags**: empirically it has been found that waste can be enveloped around plastic bags, probably to help moving it around the area or to hide them more easily

- **tubes**: metallic or plastic tubes for building use, usually found in construction yards that have been abandoned.

- **wood**: very large amount of wood placed on the ground usually indicates a possible offence, since it is illegal to store in the open large quantities of materials.

- **grouped cars**: again empirically it has been found that the presence of groups of cars in abandoned areas can indicate the presence of an illegal landfill. The possible reasons are the ditching of the vehicles or the moving of other wastes.

- **generic wastes**: wastes that have a more structured shape and that cannot fall in the other classes. It is used also when from the image the objects cannot be clearly identified.

Framing these types of relevant objects we obtained a total of 3320 annotations.

The dataset has been created with a format inspired by the COCO [37] dataset, using a preexisting web application to make the annotations of the images. Figure 4.3 shows some example of the segmentations that have been made and the derived bounding boxes that contains them. This annotated dataset is used after having trained the binary classification models to evaluate the generated CAMs.

Figure 4.3: Segmentations in blue, bounding boxes in light blue. A are scattered wastes, B pallets, C dumpsters, and D tires.

4.0.1 Data augmentation

To increase the size of the training set and be less prone to over-fitting the following data augmentation strategy has been adopted. All images in the training dataset have been resampled at different scales, to obtain for the same area 3 images of 600×600, 800×800, and 1000×1000 pixels respectively. This lead to an augmented dataset, where the same area is present 3 times but with larger or smaller contexts. The images at smaller scales also captured less context of the area.

The images were then resized into 800×800 pixels, and normalized. Finally, random flips over the horizontal and vertical axes are performed.

Chapter 5

Training the Networks

In this Chapter the various experiments and trained models are presented, detailing for each one their architectural configurations and explaining the general pipeline of the training process.

5.1 Models

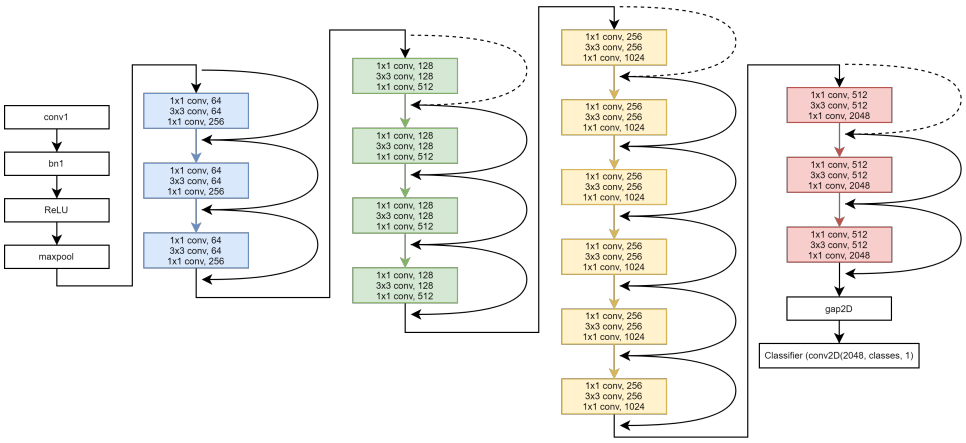


Figure 5.1: ResNet50 architecture. Coloured blocks correspond to residual blocks. Inside the blocks are indicated the kernel sizes of the convolutions and the output channels (ReLu and batch norm are omitted). The blue blocks belong to the conv2_x layer, the green ones to the conv3_x, the yellow ones to the conv4_x, and the red ones to the conv5_x.

Approaching this project, the already adopted CNN architecture for binary classification of Illegal Landfills was based on the ResNet50 architecture detailed in Section3.2. Figure 5.1 shows this model architecture. Starting from it, our aim was to study the improvements that the integration of various attention modules could bring. We opted to study and implement the

three attention mechanisms described previously in Section3.3, using the same ResNet50 architecture as the backbone model. In particular, the Convolutional Block Attention Module (CBAM [60]) implementation have been made parameterizable to be able to replicate the Squeeze-and-Excitation Module and the Efficient Channel Attention Module, making us able to perform experiments that combined various features of these modules. Each attention module is placed at the end of each convolutional bottleneck of the ResNet50, just before the endpoint of each identity shortcut connection. A simple example of how the attention modules are integrated with the ResNet50 skeleton is showed in Figure 5.2.

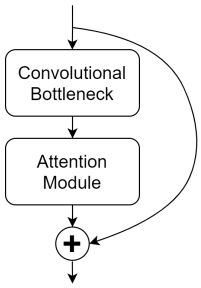


Figure 5.2: Diagram showing how the attention module is attached to the ResNet50 backbone.

At the higher abstraction level, our attention module is composed of two separate submodules, the Channel Attention module and the Spatial Attention module, that can be used one at a time or in conjunction. If it is chosen to use them both simultaneously, they can be placed in sequential order or in parallel. The complete list of possible configurations is:

- Using **SE** and **ECA**:
  - **Only Channel**: using only the Channel Attention submodule. With it we can obtain the equivalent implementations of the original SE-Net paper [23] and of the ECA-Net paper [58].
- Using **CBAM**:
  - **Only Spatial**: using only the Spatial Attention submodule.
  - **Sequential Channel→ Spatial**: using in a sequential fashion the Channel Attention submodule, followed by the Spatial Attention submodule. The overall functioning has already been explained with Equation 3.6 of Section 3.3.3. With this configuration we can obtain the equivalent implementation of the original CBAM paper.

- **Sequential Spatial→ Channel**: using in a sequential fashion the Spatial Attention submodule, followed by the Channel Attention submodule.
- **Parallel**: using both submodules in a parallel fashion, applying both to the same input feature map and summing the resulting features maps to obtain the output feature map.

Then each submodule has its own possible configurations.

Channel Module Configuration

Regarding the Channel Attention module there are two main configurations: the first one regards the choice between using a Multilayer Perceptron or a 1-dimensional convolution to learn the weights of this module. Using an MLP we could recreate the SE and the CBAM module, while using the 1D convolution we can replicate the ECA module. The motives between the choice of one or the other have been already explained in Section3.3. In general, using a 1D-convolution means that the network will have fewer parameters to train and it has been shown that it is able to capture better the local cross-channel interactions. Other than that there is the choice of the pooling mechanism used to produce the channel descriptor, which can be a single one or multiples in parallel, and where we have chosen to experiment between average pooling and max pooling. To summarize the possible choices are the following:

1. Channel submodule core function:
   - Multilayer Perceptron.
   - 1-dimensional convolution.
2. Channel submodule pooling mechanism (spatial feature aggregator):
   - Average pooling.
   - Max pooling.
   - Average and Max pooling simultaneously.

Spatial Module Configuration

Instead in the Spatial Attention submodule there is a single parameter that is the kernel size of the convolution used in order to learn the weights of the spatial channel. We experimented with spatial kernel sizes of 3 and 7. Proceeding with the experiments we observed some first clear results directly during the initial experimentation phase. The most evident one was that, as already proved in the original paper of CBAM [60], the spatial sub-module having a kernel of size 7 instead of 3 was clearly better. We

experimented with both kernel sizes in most of the configurations, and this result holds in every case. Hence, to simplify the discussion, we will show only the configurations with kernel size equal to 7.

Tested Configurations

Table 5.1 shows the set of different configurations that have been explored.

ID	Module Modified	Submodules	Channel Pooling	Channel function
1	SE	**Only Channel**	**avg**	**MLP**
2		Only Channel	max	MLP
3		Only Channel	avg & max	MLP
4	CBAM	Only Spatial	-	-
5		Channel→Spatial	avg	MLP
6		Channel→Spatial	max	MLP
7		**Channel→Spatial**	**avg & max**	**MLP**
8		Spatial→Channel	avg	MLP
9		Spatial→Channel	max	MLP
10		Spatial→Channel	avg & max	MLP
11		Parallel	avg	MLP
12		Parallel	max	MLP
13		Parallel	avg & max	MLP
14	ECA	**Only Channel**	**avg**	**1D conv**
15		Only Channel	max	1D conv
16		Only Channel	avg & max	1D conv

Table 5.1: Table showing the different experiments that have been made. The ID will be later use to identify the results. Bold ones are the orginal proposed configurations of the papers.

5.2 Training Environment and Hyperparameters

5.2.1 Training Environment

The various script used for the training, the evaluation, and the qualitative analysis have been written in Python, using mostly the PyTorch library. The experiments have been run on a Linux VM with Ubuntu 18 as the operating system, having two NVIDIA GeForce RTX 2080Ti GPUs.

5.2.2 Hyperparameters

Various hyperparameters have been used to tweak the training process, and a list of the relevant ones for the experiments that have been conducted is given below.

Number of Epochs

An epoch is an entire training passage, where all the samples in the dataset have been fed to the network once. The number of epochs is a positive integer that fixes the maximum running time of the training, halting the procedure after the given amount of training passages have been done. It's usually coupled with other methods to optimize the computational time.

Early stopping patience and minimum delta

The early stopping mechanism is used to halt the training procedure before the specified number of epochs, checking for improvements over a validation set and stopping if a criterion is not met. In particular, for each epoch, it checks if the loss function computed over the validation dataset is less than the difference between the minimum of the same metric over all the epochs and the *min delta*, that is a criterion to check whether the current training passage has brought improvements (also other metrics different from the loss function could be used). Then it counts the number of consecutive epochs for which the aforementioned criterion has not been met. If this count exceeds the *stopping patience* number, the training process is stopped, having recognized that for a number of consecutive epochs there were no substantial improvements in the model.

Batch size

An iteration of the optimization procedure of the training process is made with a subset of the training dataset. The number of the training samples that are jointly used to estimate the error with the loss function and to update the weights with gradient descent is the batch size. Hence, for each epoch, we have a total number of iterations equal to $batch\ size/N$, where N is the total number of samples in the training dataset.

Learning Rate

The learning rate is a number, usually constrained in $[0, 1]$, that is the multiplication factor of the gradient in the gradient descent algorithm. Intuitively, it determines how much the weights of the model are changing during each backward path. A common practice that followed from empirical observations is that it is useful to initialize the learning procedure with a high enough lr and then decrease it progressively during the training procedure.

Weight Decay

To prevent overfitting problems a common ML practice is to introduce a regularization directly in the computation of the loss function. In order to do it, a penalty term is added to the loss function, which effectively discourages the weights from reaching large values. There are various measures that could be used as a penalty term and for the scope of this project the commonly used L_2 norm regularization has been used. With it the resulting formula for the loss function is:

$$L_{Regularized} = L_{Data} + \lambda * L_{weights} = L_{Data} + \lambda * L_2 \tag{5.1}$$

where $\lambda \in [0, 1]$ is the weight decay.

Weight Initialization

Weights initialization is crucial to have less training time and to have a good convergence using gradient descent. As explored in Section2.3.5, transfer learning is one of the main procedures utilized in our experiments to also benefit from the advantages of transferring the knowledge gained on other classification tasks. Nonetheless, pre-trained weights were not always available, or were missing for some of the modules of the trained networks. Hence, a PyTorch implementation of the Kaiming initialization [21] (or He initialization) has been tested. Intuitively, this method initializes the weights of each layer sampling from a standard normal distribution and scaling those samples with a term that considers the number of connections that are present at that layer level. Moreover, another factor it takes into consideration is that the activation functions used in the network are $ReLu$s.

First Trainable Layer

Again as explored in Section2.3.5, while transfer learning is used, an additional technique that could be implemented is freezing some of the initial layers. The First Trainable Layer represents the first layer of the network for which the weights will be updated during the training procedure. The weights of the layers before will instead be fixed to the ones that are taken from the pre-trained model. Since all experiments are done with a ResNet50 backbone, the First trainable Layer values are constrained between $[0, 5]$.

Chapter 6

Evaluation of the Results

This Chapter presents the quantitative and qualitative analysis of the results of the performed experiments.

In particular, in Section 6.1 we evaluate the quantitative classification results. In order to do so, we present the choice of the hyperparameters that have been made in order to do the training, followed by the definitions of the performance metrics that have been used to evaluate those experiments. With this basis, we then present the numeric results of the various attention-enhanced models and compare their performances. We also propose and evaluate another variation of these attention-enhanced models, named ECA+Spatial, that combines the spatial attention module of CBAM with the channel attention of ECA. Then, using a subset of those models, composed by the networks with the original attention modules proposed by the papers [23], [58], [60] and the ECA+Spatial one that obtained the best performance, we compared and interpret their performances with the baseline ResNet50.

In Section 6.2 instead, we propose a further evaluation of the capabilities of the various models that have been built, looking at the classification results on some sample images, to understand more deeply the strengths and limitations of our models.

6.1 Classification quantitative evaluation

6.1.1 Experiments

As already seen in Section 5.1 and in particular in Table 5.1 we have focused our classification experiments on different architectures. For each one of them, various trainings with different values for the hyperparameters described in Section 5.2.2 have been run to optimize the training. Usually, for a given architecture and a given set of hyperparemeters, multiple trainings have been made, between 2 and 5, in order to better estimate the performance of the given training configuration. This is due to the fact that

some random elements could influence the outcome of the training procedure, such as the random order in which the training samples are fed to the network. Some hyperparameters like the number of epochs, the early stopping parameters, and the batch size have been mostly fixed a priori, noticing from the very first trainings that there were clear values for which the training outcome was better or the training procedure was faster without additional downsides. In particular for the **number of epochs** has been fixed at 80, seeing that most of the time the early stopping mechanism intervened before this stopping criterion was met. The **early stopping patience** and the **minimum delta** have been mostly fixed around 10 and 0.005 respectively, seeing from the curves of the training loss compared to the validation loss (over the epochs) that with those values the trainings were stopped when the network started to overfit. The **batch size** also has been fixed at 14, seeing empirically that increasing it was beneficial to the performances, and 14 being the highest value that the GPUs memory could handle without overflowing. Two hyperparameters that were instead tweaked to search for an optimum are the learning rate and the weight decay. For the **learning rate** we have seen early during the first experiments that the best values were constrained by $(10^{-2}, 10^{-4})$. Too high values like 10^{-2} prevented the convergence, while too low values like 10^{-4} led to a slow convergence that required too much time and required a relaxation of the early stopping. We settled to use for most of the experiments the values $\in \{0.0075, 0.005, 0.001, 0.00075, 0.0005\}$. Following a similar path, looking first at different orders of magnitude and then selecting the most performing range, for the **weight decay** we used values $\in \{0.001, 0.0005, 0.0001\}$.

Regarding **weights initialization**, three sets of pre-trained weights, one for each attention module, have been found publicly available online [19][47][61], all pre-trained on the ImageNet [14] dataset. Hence, when the architectural models were the default implementation of the attention modules, those full pretrained weights have been used. Instead, for the other experiments where there were changes in the architecture of the modules, various strategies have been tested:

1. using the attention modules downloaded pre-trained weights
2. using the attention modules downloaded pre-trained weights or the ResNet50 pretrained weights (again trained on ImageNet) only for the backbone, and for the attention modules initialize with:
   (a) the default PyTorch initialization
   (b) the Kaiming (He) [21] initialization

Among these strategies, the best results have been obtained using the ResNet50 pretrained weights for the backbone, and the Kaiming initialization for the attention modules. Moreover, given these initialization mechanisms, the

freezing technique has been applied only when all the weights were initialized with the pre-trained ones. So, when it was possible to use this technique, we tested with **First Trainable Layer** values in $[0, 3]$, having noticed from the very first experiments that higher values led to really bad performances. The results obtained, which will be shown later below, were that the freezing technique is in fact beneficial to the performances, having obtained some of the best results by freezing the first 2 and 3 layers.

6.1.2 Performance Measures

To evaluate the performances of our models we used a framework that is widely used in Machine Learning that is based on the **confusion matrix**, which states the number of samples that have been correctly classified against the number of samples that have been misclassified. Table 6.1 shows an example of such confusion matrix in a binary classification problem composed of two mutually exclusive classes, the Positive and the Negative one. Applying it to our task, the positive class corresponds to the images that contain an Illegal Landfill while the negative one to the images that are not an IL. In particular, the elements of the matrix in our task corresponds to

	Actual Class: Positive	Actual Class: Negative
Predicted Class: Positive	**True Positive (TP)**	**False Positive (FP)**
Predicted Class: Negative	**False Negative (FN)**	**True Negative (TN)**

Table 6.1: Confusion Matrix

the number of samples that have the following properties:

- **TP**: the model correctly predicts the presence of an IL and the image contained an IL. Equivalent to Hit.
- **TN**: the model correctly predicts the absence of an IL and the image did not contain any IL. Equivalent to correct rejection.
- **FN**: the model predicts that there is no IL where actually there was one. Equivalent to miss.
- **FP**: the model predicts that there is an IL where actually there was not one. Equivalent to false alarm.

Based on this values the following metrics can then be evaluated.

Accuracy

$$Accuracy = \frac{TP + TN}{N} \tag{6.1}$$

where N is the total number of samples. It represents the fraction of the samples that have been correctly classified in the dataset.

Precision

$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$

It represents the fraction of samples that have been correctly classified in the positive class among all the ones that have been classified as positive. In our case, it is the percentage of the samples classified as IL that are actually an IL.

Recall

$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

It represents the fraction of samples that have been correctly classified in the positive class among all the ones belonging to the positive class. In our case, it is the percentage of the IL found by the model among all the IL in the dataset.

F1 score

$$F1 = \frac{2 * Pre * Rec}{Pre + Rec} \tag{6.4}$$

It is the harmonic mean between the Precision and the Recall. It is useful since these two metrics are usually in trade-off and for this reason F1 can be more easily used to compare different models.

Average Precision

As seen, Precision and Recall are in a trade-off, and Average Precision is another measure to summarize this trade-off. In particular, given a set of thresholds, the two metrics R_t (Recall) and P_t (Precision) are computed. Then we assign a set of weights for each precision, that are made of the increase in the recall from the previous threshold ($R_t - R_{t-1}$). Finally, we then take the mean of the precisions weighted by these factors. The resulting formula is:

$$AP = \sum_{thresholds} (R_t - R_{t-1}) P_t \tag{6.5}$$

6.1.3 Results

Between all the experiments that have been made, in this Section the best ones are presented. In order to evaluate a trained model we first compute the performance metrics over the validation dataset for each threshold $t \in [0, 1]$ with a step of 0.005. Then, between all the values, the threshold corresponding to the highest $F1$ is selected (another common practice is to select the highest Accuracy, but we are more interested in the Precision and

Recall metrics, summarized by F1). With this value, we then compute the performance metrics over the testing set.

Table 6.2 summarize the experiments. For each explored architecture the performances of the best training obtained are shown.

ID	Th	Validation					Testing				
		AP	Acc	Pre	Rec	F1	AP	Acc	Pre	Rec	F1
1 (SE)	**0.3**	**0.938**	**0.918**	**0.864**	**0.891**	**0.877**	**0.930**	**0.908**	**0.826**	**0.896**	**0.860**
2	0.40	0.932	0.910	0.927	0.789	0.852	0.926	0.896	0.899	0.755	0.821
3	0.50	0.937	0.905	0.832	0.891	0.860	0.933	0.905	0.825	0.887	0.855
4	0.51	0.933	0.915	0.868	0.875	0.872	0.931	0.911	0.845	0.877	0.861
5	0.39	0.916	0.899	0.845	0.852	0.848	0.914	0.893	0.830	0.830	0.830
6	0.58	0.912	0.892	0.816	0.867	0.841	0.904	0.902	0.861	0.821	0.841
7 (CBAM)	**0.39**	**0.935**	**0.918**	**0.858**	**0.898**	**0.878**	**0.934**	**0.917**	**0.861**	**0.877**	**0.869**
8	0.55	0.902	0.902	0.836	0.875	0.855	0.914	0.887	0.809	0.840	0.824
9	0.32	0.913	0.899	0.835	0.867	0.851	0.907	0.887	0.809	0.840	0.824
10	0.44	0.904	0.894	0.809	0.891	0.848	0.908	0.884	0.786	0.868	0.825
11	0.51	0.949	0.918	0.887	0.859	0.873	0.940	0.917	0.906	0.821	0.861
12	0.49	0.945	0.920	0.922	0.828	0.872	0.942	0.914	0.897	0.821	0.857
13	0.37	0.949	0.905	0.818	0.914	0.863	0.939	0.905	0.808	0.915	0.858
14 (ECA)	0.48	**0.947**	**0.923**	**0.883**	**0.883**	**0.883**	**0.944**	**0.923**	**0.877**	**0.877**	**0.877**
15	0.20	0.931	0.892	0.803	0.891	0.844	0.921	0.890	0.800	0.868	0.833
16	0.50	0.938	0.912	0.856	0.883	0.869	0.931	0.899	0.821	0.868	0.844

Table 6.2: Best results of the different experiments that have been made. The architectural configuration can be traced back in Table 5.1 using the IDs. IDs 1-3: variations of SE. ID 4: only spatial module. IDs 5-13: variations of CBAM. IDs 14-16: variations of ECA.

Experiments with IDs from 1 to 3 are the alterations of the original SE module with different pooling mechanisms. Among those three, the one that obtained the best results in terms of higher F1 is ID 1, that correspond to the originally proposed configuration of the module with average pooling.

Looking at IDs from 5 to 13, which correspond to alterations of the sequencing of the two sub-modules and of the pooling mechanism, it can be seen that the CBAM configuration suggested from the original authors of the paper, corresponding to ID 7 that is the one with sequence channel→spatial and average&max pooling, is again the best configuration considering F1. The configurations that parallelize the two sub-modules, IDs 11-13, seem promising, but still leading to a lower F1 compared to ID 7. Also the experiment using only the spatial module, ID 4, have worst performance than the best of CBAM, reaffirming that in this case is better to use the two modules jointly.

Comparing the IDs from 14 to 16, that corresponds to alterations of ECA, ID 14 is the best performing architecture, obtaining the highest AP, Accuracy and F1. As in the case of SE, altering the pooling mechanism (IDs 15,16) does not lead to better performances.

Following these results, seeing that ECA's channel module performs better than the SE architecture and that the addition of a spatial module to SE, that is CBAM, leads to better performances, we tried to experiment combining both these improvements, replacing the channel module inside CBAM with the ECA module (and altering the channel pooling mechanism). We called this module ECA+Spatial and the experimented configurations are shown in Table 6.3.

ID	Module Modified	Submodules Sequence	Channel Pooling
17	ECA+Spatial	Channel→Spatial	avg
18		Channel→Spatial	max
19		Channel→Spatial	avg & max
20		Spatial→Channel	avg
21		Spatial→Channel	max
22		Spatial→Channel	avg & max
23		Parallel	avg
24		Parallel	max
25		Parallel	avg & max

Table 6.3: Different configurations of the ECA+Spatial module.

The results of these experiments are shown in Table 6.4. It can be seen that parallelizing the sub-modules (IDs 23-25) is the best strategy. To understand which one is the best configuration among those, we looked at the validation performances, that would suggest to select ID 25 (average max pooling), given the higher accuracy and F1.

ID	Th	Validation					Testing				
		AP	Acc	Pre	Rec	F1	AP	Acc	Pre	Rec	F1
17	0.51	0.915	0.899	0.868	0.820	0.843	0.913	0.887	0.847	0.783	0.814
18	0.54	0.899	0.897	0.828	0.867	0.847	0.896	0.878	0.793	0.830	0.811
19	0.48	0.902	0.879	0.796	0.852	0.823	0.880	0.878	0.793	0.830	0.811
20	0.47	0.909	0.892	0.821	0.859	0.840	0.897	0.872	0.752	0.887	0.814
21	0.48	0.915	0.899	0.850	0.844	0.847	0.888	0.878	0.828	0.774	0.800
22	0.25	0.910	0.881	0.793	0.867	0.828	0.883	0.875	0.776	0.849	0.811
23	0.37	0.941	0.915	0.863	0.883	0.873	0.935	0.917	0.868	0.868	0.868
24	0.39	0.952	0.920	0.844	0.930	0.885	0.939	0.917	0.820	0.943	0.877
25	0.32	**0.952**	**0.930**	**0.891**	**0.898**	**0.895**	**0.944**	**0.917**	**0.890**	**0.840**	**0.864**

Table 6.4: Best results of the different experiments that have been made on the ECA+Spatial module.

In this case we also show another tool that was used to compare the classification performances, the **Precision-Recall curve**. This curve helps

in understanding the trade-off made by the model between the Precision and the Recall, plotting these two measures for a complete set of thresholds. Figure 6.1 shows this curve computed for those 3 experiments, using the threshold values $t \in [0, 1]$ with a distance of 0.005.

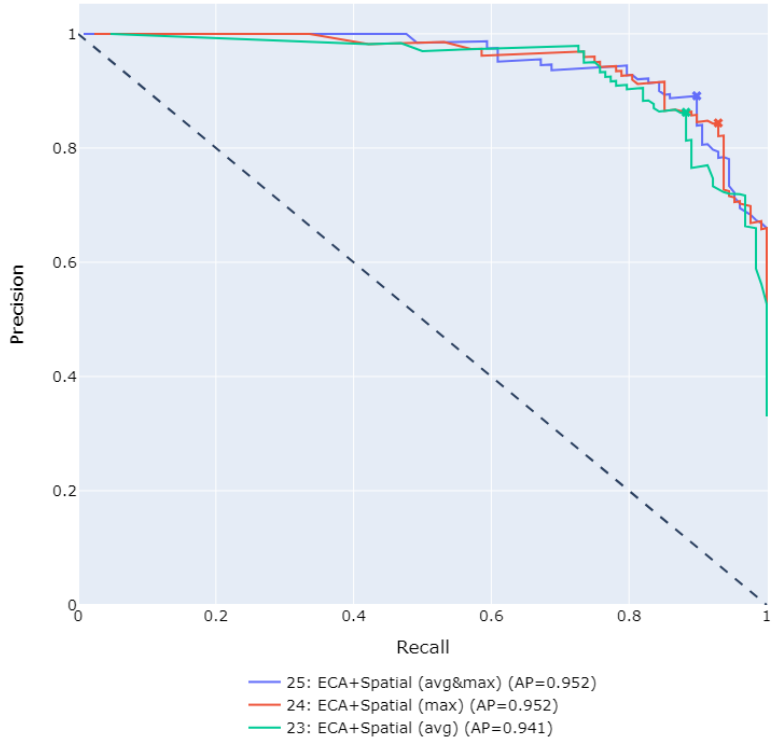


Figure 6.1: Comparison between the precision-recall curves of ECA+Spatial with Parallel sequencing and different pooling mechanisms. The markers on the curves represent the points at the threshold defined in Table 6.5 (higher validation F1).

Looking at the precision-recall curves from Figure 6.1 we are mostly interested in the top right part of the graph, that corresponds to high values of precision and recall simultaneously. Between ID 25 and 24 we preferred the first one, being less skewed towards the recall (confirmed also from the results in Table 6.2). This in combination with the analytical results previously mentioned confirmed our choice of the experiment with ID 25.

It has been shown the comparison between the various configurations for each attention module architecture. It follows the comparison between the original versions of the three attention modules, the best performing model with ECA+Spatial and the already standalone implementation of

the ResNet50 architecture (from here on also called ResNet for simplicity). Table 6.5 shows the classification performances of these models.

ID	Th	Validation					Testing				
		AP	Acc	Pre	Rec	F1	AP	Acc	Pre	Rec	F1
SE	0.3	0.938	0.918	0.864	0.891	0.877	0.930	0.908	0.826	**0.896**	0.860
CBAM	0.38	0.935	0.918	0.858	0.898	0.878	0.934	0.917	0.861	0.877	0.869
ECA	0.48	0.947	0.923	0.883	0.883	0.883	0.944	**0.923**	0.877	0.877	**0.877**
ECA+Spatial	0.32	0.952	0.930	0.891	0.898	0.895	0.944	0.917	**0.890**	0.840	0.864
ResNet	0.4	0.946	0.920	0.888	0.867	0.877	**0.946**	0.917	0.882	0.849	0.865

Table 6.5: Classification performance comparison between ResNet, the three default attention modules, and the ECA+Spatial attention module.

Looking at these performances, there are two results that draw our attention. The first one being that SE was not able to improve the ResNet performances in this context, as it did instead in the original paper on the ImageNet classification task (see Section 3.3.1). The second is that our intuition that lead to the design of ECA+Spatial did not increase the performances with respect to ECA.

Looking at the metrics we are mostly interested in the precision and recall, since they dictate how well the generated models are able to make correct IL alerts, and how many IL are found among the ones in the data. Hence we look at the F1, that presents the trade-off between the previously mentioned metrics, and the AP, that summarizes the area under the precision-recall curve. From there it can be seen that ECA is the model that generalize better, which means that it maintains the more similar performances passing from the validation to the testing set. Analytically, ECA loses only 0.36% in AP and 0.62% in F1, while the mean loss between all the models is 0.43% for AP and 1.68% for F1.

This said, ECA is overall the best model, obtaining significant performance gain over ResNet and being the most balanced model between precision and recall. This result is also coherent with the results in the literature, where ECA is the best attention module compared to SE and CBAM for the ImageNet classification task. Finally, with an F1 of 0.877 and an Accuracy of 0.923, ECA proves to be an architecture able to perform well this task of Illegal landfills scene classification.

6.2 Classification qualitative evaluation

In order to obtain a better understanding of the results shown in Table 6.5, we looked directly at the samples of the testing set, starting from those where the various models disagreed on their prediction.

Between all the images inside the testing set, we first started analyzing the positive samples (the ones containing an illegal landfill). Of them, $\sim$75%

were recognized as true positives by all the models. Looking at many of the examples that have been correctly predicted by most models, a particularly illustrative example is given in Figure 6.2. There it can be seen that SE-Net, CBAM, and ECA are correctly classifying this sample. The CAMs of these models indicate that their prediction is given exactly thanks to elements in the image that characterize the IL. Instead, ResNet and ECA+Spatial are the two models that miss-classify this image. However, between these two, ECA+Spatial has more precise CAMs, and the region considered important further down is characterized by the presence of shadows, an element that often creates problems for all models. On the other hand, ResNet does not fail to see the relevant objects, but is misled by many other elements that are completely irrelevant.

Figure 6.2: Image of a positive sample correctly predicted by most models (CAMs produced with threshold t=0.25). In the original image, the yellow box indicates a set of scattered wastes, the red one a set of pallets and IBC.

For ∼5% of the positive samples, all the models miss the IL. A characteristic that leads to this error regards the spatial extension of the IL. Figure 6.3 shows one of the samples of IL that all the models fail to classify. This example is characterized by a noticeable amount of scattered wastes (in white, nearby the crossing of the roads) and even if that particular is deemed relevant by the CAMs of all the models (omitted in the picture), the prediction is always negative. This could be due to two main reasons, the first one being that the extension of the visible wastes is contained, and the second

one is that this place is subject to a very advanced state of abandonment, where nature has covered most of the buildings. In fact, usually ILs are found nearby industrial areas or in nature but in open fields, and this could lead the models to give less relevance to the waste objects that they found on this scene. Another relevant example is shown in Figure 6.4, where an IL sample is given by dumps of scattered wastes that from the aerial image are mostly covered by shadows. As already stated their presence proves to be a challenge and here the models struggle to recognize the objects inside the shade of the building. As counterproof, the small part outside of the shadow is recognized to be relevant by the CAMs but nonetheless, since most of the IL is hidden for all the models, for all of them this sample is another False Negative.

Figure 6.3: Illegal landfill with few visible wastes, that resulted in a FN for all the models. On the right the CAMs produced by ECA+Spatial with $t = 0.15$ (other models have similar CAMs).

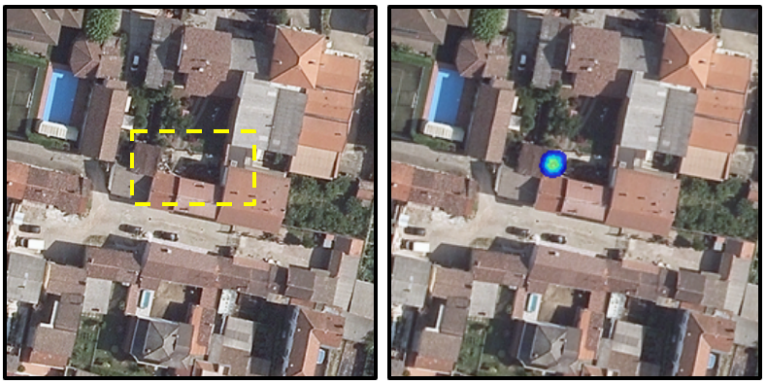


Figure 6.4: Illegal dumping with scattered wastes mostly covered by shadows, indicated by the yellow box. On the right the CAMs produced by ECA with $t = 0.25$ (other models have similar CAMs).

Going instead to look at the negatives samples of the testing set, the same analysis of many images have been conducted looking at the False Positives and at the True Negatives. Overall ∼2% of these negative samples are erroneously classified by all the models, and we propose two of them as examples, to explain a characteristic that could lead the classifiers to make a wrong prediction. This examples are shown in Figure 6.5. As it can be seen, the CAMs were all focused on the stored wood. Those objects are in fact one of the relevant ones that could signal the presence of an IL, but in these examples are probably only placed outside for storage, without a clear indication of other wastes or dangerous materials nearby. The example on top can also emphasise again the difficulty in recognizing objects under a shadow.

Figure 6.5: Examples of False Positives misclassified by all the models. On the right the CAMs produced by ECA with $t = 0.3$ (other models have similar CAMs).

Moreover, in Figure 6.6 we present two other examples, that show negative samples that have been misclassified by ECA. In these cases there is no visible object that could be present also in an IL, but the terrain is the element that misled the correct prediction, as it can be seen looking at the CAMs. Again, is not unusual to find ILs over these types of terrain.

Figure 6.6: Negative samples (not IL) with a quarry and a place with rough terrain. On the right the CAMs produced by ECA with $t = 0.2$.

Chapter 7

CAMs Analysis

In the following Chapter we will see another evaluation approach that highlights additional benefits of these attention mechanisms.

In the first section, we make a qualitative evaluation, using the annotated dataset described in Chapter 4 to compare the Ground Truths with the various Class Activation Maps (CAMs) generated by the baseline ResNet and the attention networks.

In the second section, we give a qualitative evaluation, looking at some example images, to highlight some strengths and weaknesses of the CAMs as a tool for object detection.

7.1 CAMs quantitative evaluation

Using the models of Table 6.5, the comparison between their produced CAMs has then been studied.

Given a Class Activation Map, that is a matrix with dimensions equals to the image dimensions and values in $[0, 1]$, and placing a threshold $t \in [0, 1]$, the whole image can be separated in a foreground area (pixels with CAM value greater or equal to t) and background area (pixels with CAM value lower than t). This means that with the threshold value t it is possible to generate a CAM *mask*, that corresponds to the foreground area and should represent the regions of the image that the model thinks are important to classify the whole image as illegal landfills. Variations to the threshold t result in smaller or wider regions being highlighted and different areas deemed relevant for the IL detection. Figure 7.1 shows an example of this behaviour. To evaluate the goodness of the CAMs generated by the various models, the resulting CAM mask, generated with thresholds ranging from 0 to 1 with a 0.05 step, are compared with the Ground Truth (GT) segmentations that are available in the annotated dataset. We propose and show the results for 4 different metrics useful to evaluate different proprieties of the CAMs.

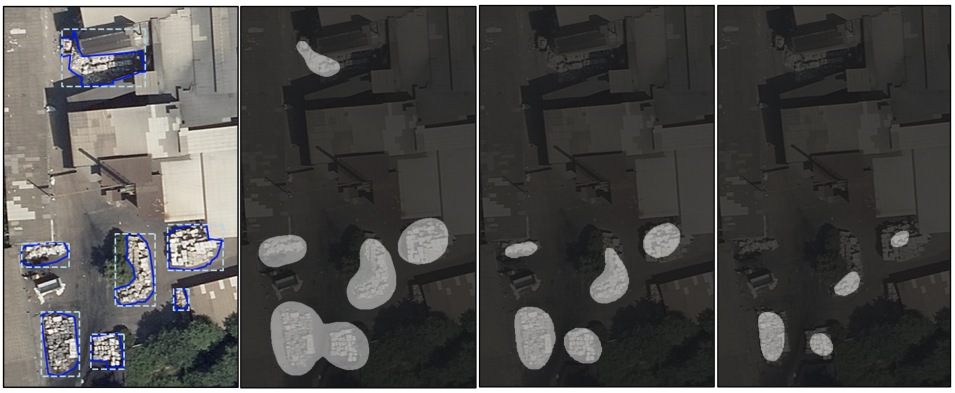


Figure 7.1: Effects of the variation of the threshold on the generated CAM mask (light grey foreground areas).

Component IoU

The first introduced metric is based on the standard evaluation metric of Intersection over Union (IoU), which is commonly used to compute the overlap between image regions. The standard definition is:

$$IoU = \frac{A_{\cap}}{A_{\cup}} \tag{7.1}$$

where $A_{\cap}$ represents the intersection between the two areas and $A_{\cup}$ the union. It follows that $IoU \in [0, 1]$, with 0 corresponding to two areas completely disjointed and 1 with areas with the same dimension and perfectly overlapping.

In particular, **Component IoU** is used in this analysis to evaluate how well the CAMs focuses on the individual relevant objects (the GTs) in the scene. It is computed by first dividing the CAM mask into *connected components*, that are the groups of pixels in the foreground area that are connected to each other. Then the IoU value is computed between *each ground truth mask* and *all the connected components that intersect it.* If a connected component does not intersect with any ground truth mask it is given an IoU equal to zero. Afterward, the average IoU across all the ground truths is taken, obtaining the overall Component IoU over the whole image.
Figure 7.2 shows the results of applying the Component IoU metric using the CAMs produced from the five CNNs under comparison. To initially understand the graph it can be seen that for all the curves, the metric increases continuously and the range of the best performance is $t \in [0.25, 0.45]$. This behaviour is motivated by the following facts: for small values of t the CAMs tend to be wider than the annotations, hence while increasing t the intersection remains similar while the union decreases, resulting in an increase of

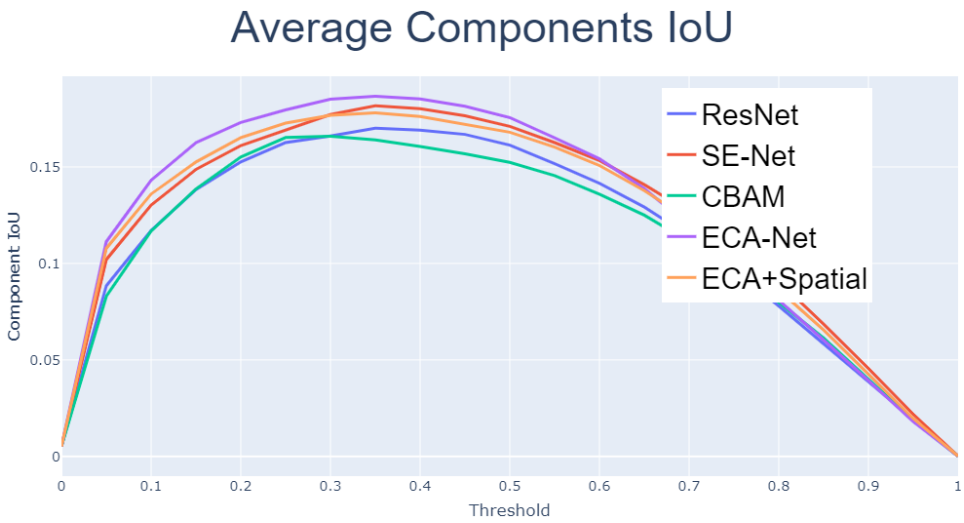


Figure 7.2: Component IoU graph over the 20 evenly spaced thresholds for the five CNNs under comparison.

the IoU values. This stops after a certain threshold (around 0.45) since the CAMs have become smaller than the GTs, hence the intersection begins to shrink and the IoU decrease towards the final value of zero for $t = 1$.

The most performing method under this metric is ECA-Net. This is due to the fact that the CAMs produced by ECA-Net are more focused on the annotations areas and usually the CAM components overlap with more of the GT masks with respect to the other models. In fact, the others tend to produce larger areas and less separate components. This leads to an increase in the size of the Union for many GTs that is not compensated by an equivalent increase of the intersection, leading to lower IoU values. An example of this behaviour can be seen in Figure 7.3 where, for a sample image, the component IoU values obtained at threshold 0.25 are shown. In it, the GT masks are denoted by the red shapes, while the contour areas are the CAMs mask, with blue being the area with values closer to 0.25 and red/white being the ones closer to 1.

As it can be seen, the model with the highest value for the metric is ECA-Net, closely followed by ECA+Spatial. Then there is SE-Net, CBAM and the final one is ResNet. This result derives from the fact that ECA covers well enough most of the GTs with better separated connected components. ECA+Spatial seems to cover a little bit better the leftmost GT, but loses something in a lower coverage of the bottom-left one and covers 3 separated GTs with a single connected component on the top-right. SE-Net covers well the GTs, but again some components are not separated. The same issue could be seen also for CBAM and ResNet, which also have some CAMs going outside of the GTs borders.

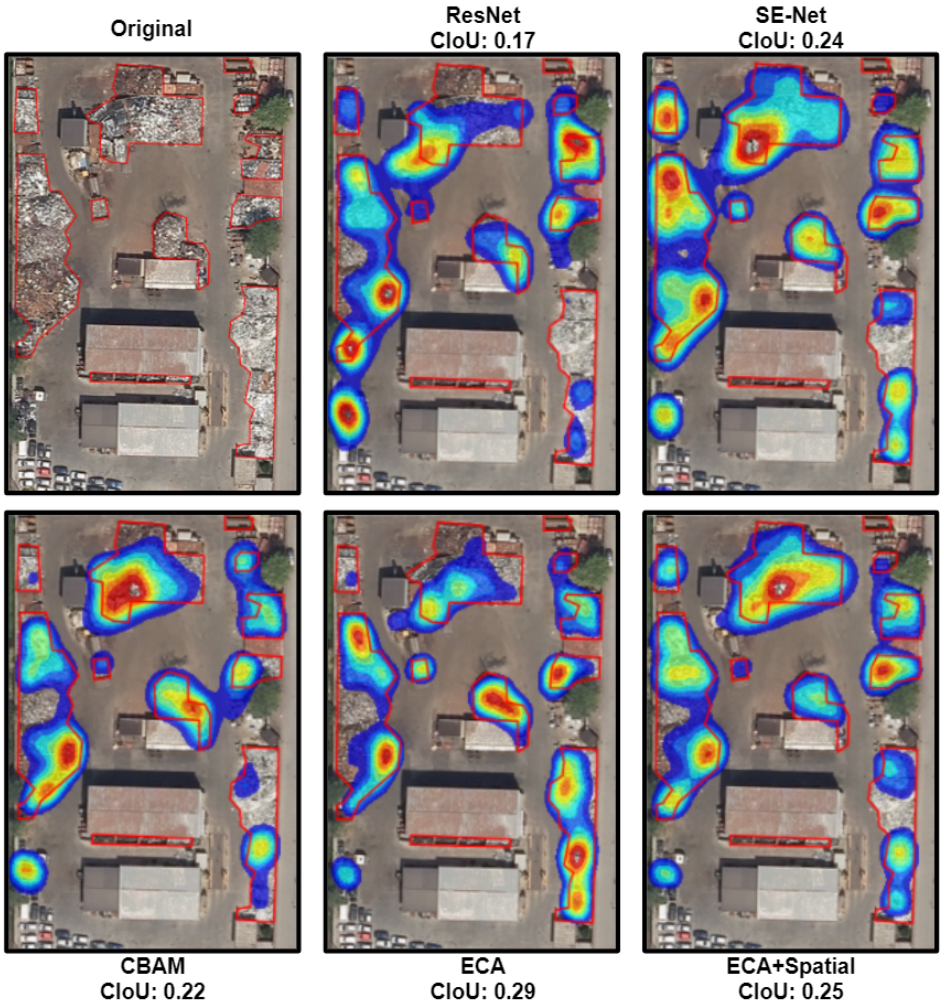


Figure 7.3: Sample for which the CAMs have been produced at threshold t=0.25. The different values of the Compontent IoU metrics are shown.

Global IoU

An alternative metric that focuses on how the entire CAM focuses on the whole representation of the waste dump scene is the **Global IoU**. It is computed by calculating the IoU between the *union of all the Ground Truth masks* in the image and *the entire foreground area* of the CAM, taken at a given threshold. In general, with respect to the Component IoU, it penalizes less those Class Activation Maps with wider and more connected areas. From Figure 7.4 it can be seen that in general for all the models the global IoU is $\sim 10\%$ higher than the component IoU.

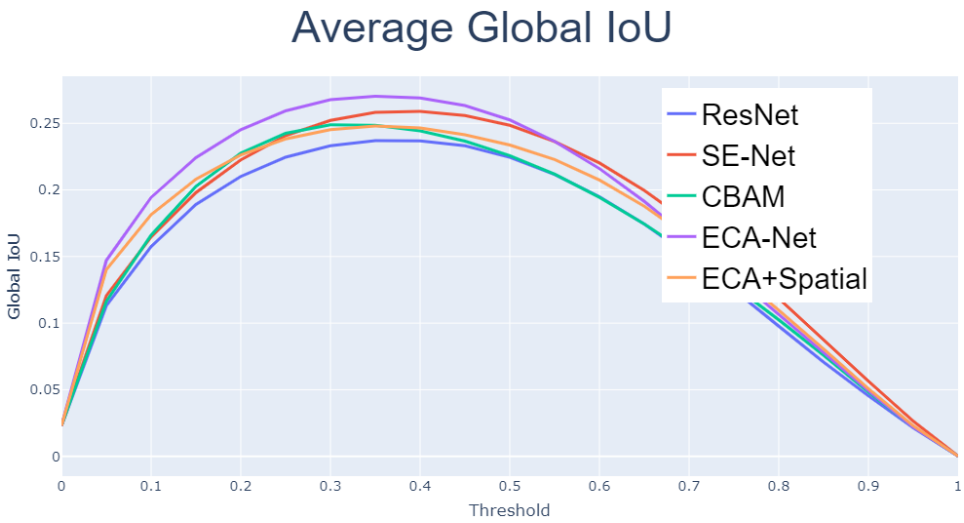


Figure 7.4: Global IoU graph over the 20 evenly spaced thresholds for the five CNNs under comparison.

This is derived from the fact that CAMs that intersects more than one component penalize a lot less the metric, having an increase in the union term that is much smaller. While the best method remains ECA-Net followed by SE-Net, CBAM is the method with the higher improvement. Compared with the Component IoU metric, where ResNet outperformed CBAM, here with the Global IoU CBAM performs better than ResNet, reaching for threshold values lower than 0.3 the same performances of SE-Net. ECA+Spatial on the other hand loses a lot compared to the other IoU metric. With the Component IoU, it holds that SE-Net and ECA+Spatial were really similar in the performances, while with the Global this holds only for low threshold values. From thresholds higher than 0.3, with the global metric ECA+Spatial is outperformed by SE-Net.

Figure 7.5 is a valid example, where the CAMs have been taken for a threshold value of $t = 0.25$, to compare this metric between the models. Even if the differences are not too wide, ECA outperforms the others thanks to a CAM that covers almost all the areas of the GTs. CBAM and SE-Net instead are shown as having an equal value of global IoU, but it can be easily seen that the motifs are different. CBAM struggles to cover some GT, leading to a lower intersection term, but also produces a smaller CAM outside the GTs' areas, leading to a lower union term. Instead, SE-Net CAMs cover more area of the GTs but also increases the union term by a contained but relevant factor, having a single larger connected CAM. ECA+Spatial performances seem to be affected by the same issues of SE-Net, with a similar uncovered area and almost the same area outside of the GTs borders. Finally, ResNet is the one with the most CAM area outside

of the GTs, considering also the 2 top-left circles, and hence is the one with the lowest performance.

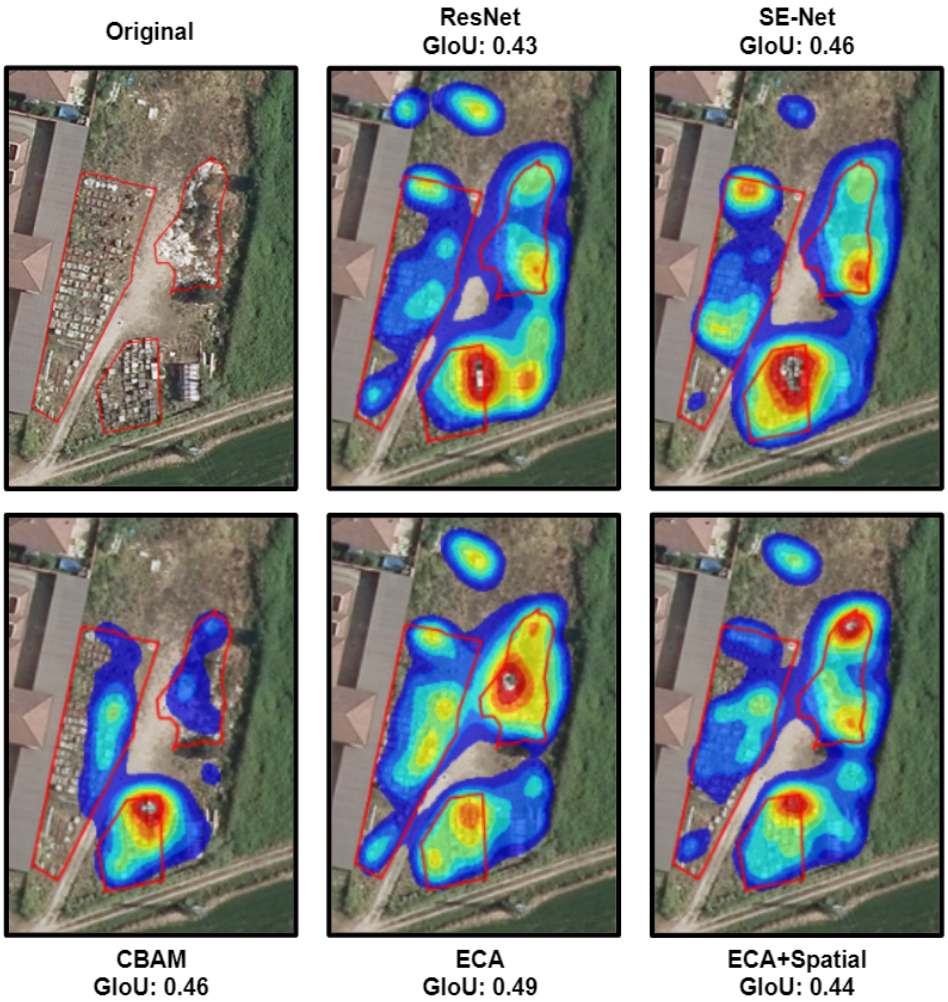


Figure 7.5: Sample for which the CAMs have been produced at threshold t=0.25. The different values of the global IoU metrics are shown.

Annotation coverage

This metric assesses the percentage of Ground Truth masks that are covered by the class activation maps. It is computed with an additional parameter, the coverage threshold $t_{cov} \in [0, 1]$, that indicates the percentage of the GT's area that needs to be covered by the class activation map foreground area in order to consider that GT's mask is covered by the network. In the following analysis and results we consider a constant threshold coverage equal to 30%. This metric is especially useful to highlight the capacity of the

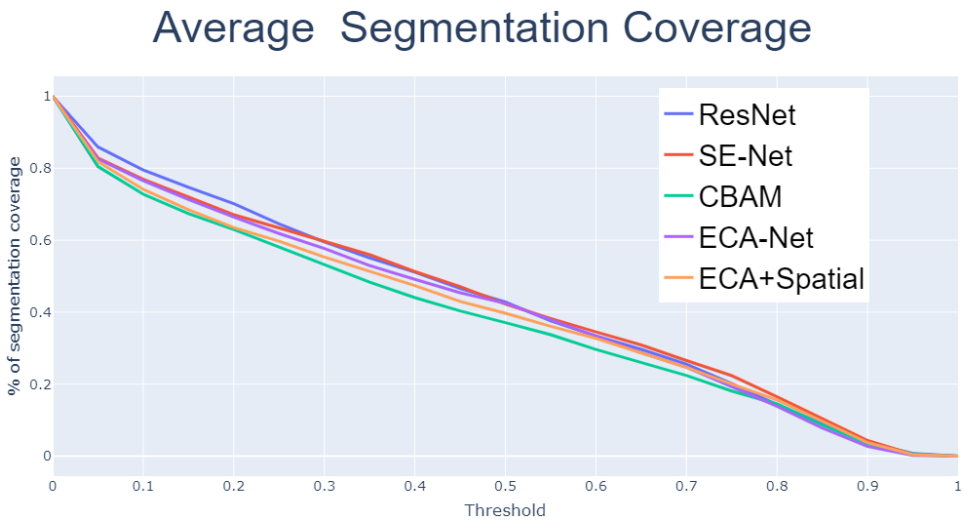


Figure 7.6: Graph of the Annotation Coverage over the 20 evenly spaced thresholds, considering a coverage threshold of 0.3, for the five CNNs under comparison.

trained models to identify a significant fraction of the relevant objects that have been recognized by the experts, but could be easily misleading. In fact, taking a trivial activation map that overlays the whole image would always return 100% coverage, even if it is the worst possible case. Nevertheless, it is still useful if coupled with the other metrics, because excessively high values of this one would be easily understood when looking at the others. Figure 7.6 shows the coverage curves for the compared models.
As it can be noticed, the models have really similar results on the coverage, except CBAM and ECA+Spatial (the latter only for the lowest thresholds), which performs slightly worse. We have already seen in the example Figure 7.5 that CBAM tends to struggle from time to time to cover some GT, and the lower value for Component IoU gives another evidence that this method tends to generate CAMs that are smaller and that do not extends to all the relevant objects. Figure 7.8 is an example of this behaviour, where CBAM and ECA+Spatial are shown to have the worst coverage compared to the other models.

Overall from the curves in Figure 7.6 it can be deduced that using a t_{cov} of 0.3 and $t \in [0.25, 0.45]$, that is the range of values corresponding to the higher Component and Global IoUs, it is possible to obtain a coverage of the annotations between 60% and 40%. This indicates that the CAMs could be a good method to localize suspicious objects inside the image.

Irrelevant attention

The final metric is the **Irrelevant Attention** and aims to measure how much the class activation maps focus on irrelevant parts of the image. It

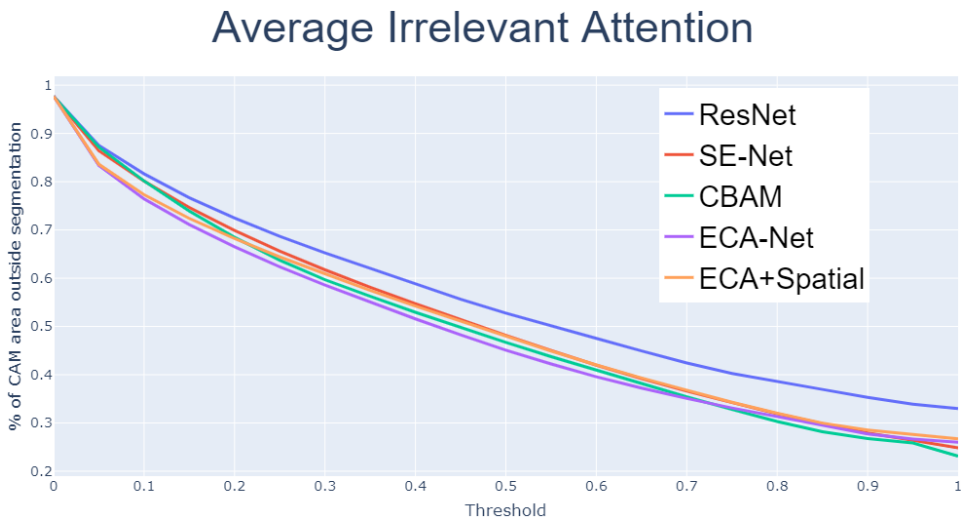


Figure 7.7: Graph showing the percentages of Irrelevant Attention over the 20 evenly spaced thresholds for the five CNNs under comparison.

corresponds to the percentage of the Class Activation Map area that does not intersect with any Ground Truth annotation. It is useful to glue together all the other metrics, helping to explain low IoU values. This is because the reasons behind a low IoU could be traced back to:

1. Small intersection area between the Ground Truth mask and the Class Activation Map foreground area

2. The two maps overlaps really well, but one of them is much larger than the other

In the first case, we would have a low value of Irrelevant Attention, while the second one could be identified by large values of irrelevant attention if the CAMs are wider than the GTs. Figure 7.7 shows the last graph of the trends that this metric follows for each one of the compared models. ECA once more is the best performing model, having the least irrelevant attention. This could be interpreted as another indicator of the ability of this architecture to focus on the relevant objects in the images. On the other hand, we have ResNet as the model with the highest irrelevant attention. This result is particularly relevant, since it can be thought as proof that the attention mechanisms, in general, improve the ability to focus more on the relevant parts of the inputs, as it has been shown looking at the produced CAMs of the models with and without an attention module.

Figure 7.8 is an example image and compares the irrelevant attention values obtained with our five models. As expected ResNet is the worst performing model, and the reasons behind this result are a medium-sized area signaled by the CAM on the right that does not correspond to any GT, and a single

large connected area that covers the four GTs in the middle. SE-Net and CBAM perform equally even if also SE's CAM signal that irrelevant area on right. However SE covers more GTs (higher coverage), so computing the ratio of irrelevant CAMs the irrelevant one on the right weights less. CBAM having some coverage problems leads instead to irrelevant areas that weights more when computing the ratio. ECA-Net is performing similar to the two last mentioned models, having good coverage and focused areas around the covered GTs, if not for the leftmost part where the focus is also outside of the GT border. Finally, ECA+Spatial is the best performing model in this example, but once again combining the two metrics can lead to further understanding. In fact, even if the Irrelevant Attention is contained, the Coverage is worst than the other models. All in all, it suffers from the same problems of CBAM, but thanks to the CAMs being a bit more focused over the GTs regions, the Irrelevant Attention results are lower.

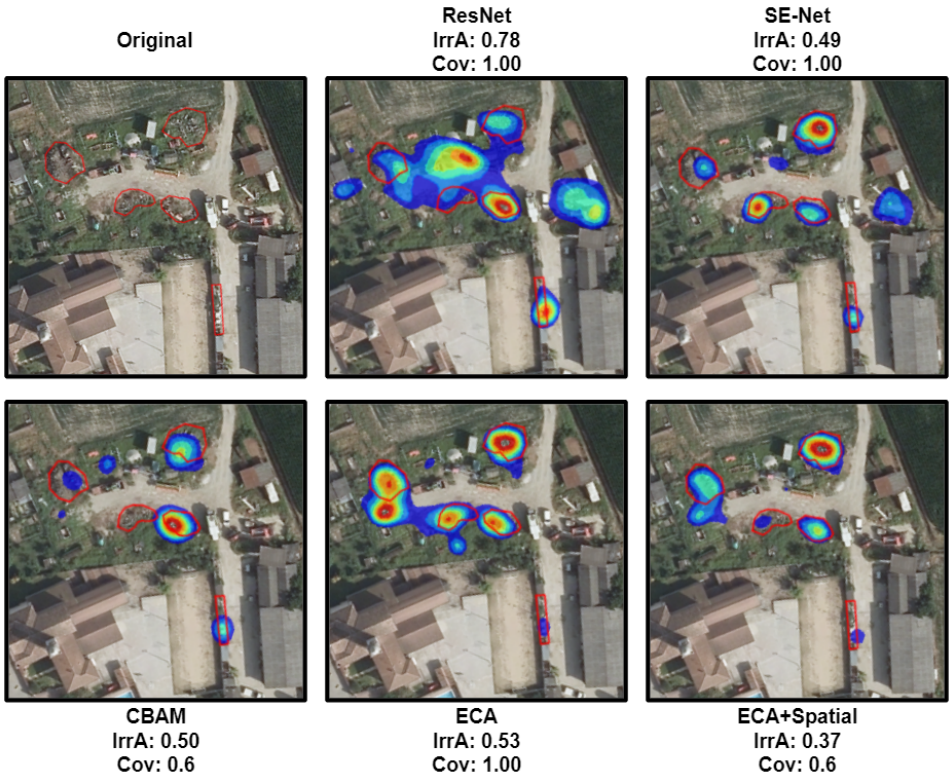


Figure 7.8: Sample for which the CAMs have been produced at threshold t=0.35. The different values of the Irrelevant Attention (and Annotation Coverage) metric are shown.

7.2 CAMs qualitative evaluation

A qualitative analysis of the class activation map is shown with the help of illustrative images in order to understand their strengths and limitations in the detection of the relevant objects inside the scene.
First of all, since we stated that all the networks were able to cover a good amount of ground truths, we propose the example in Figure 7.9, in which both objects of different shapes have been identified.

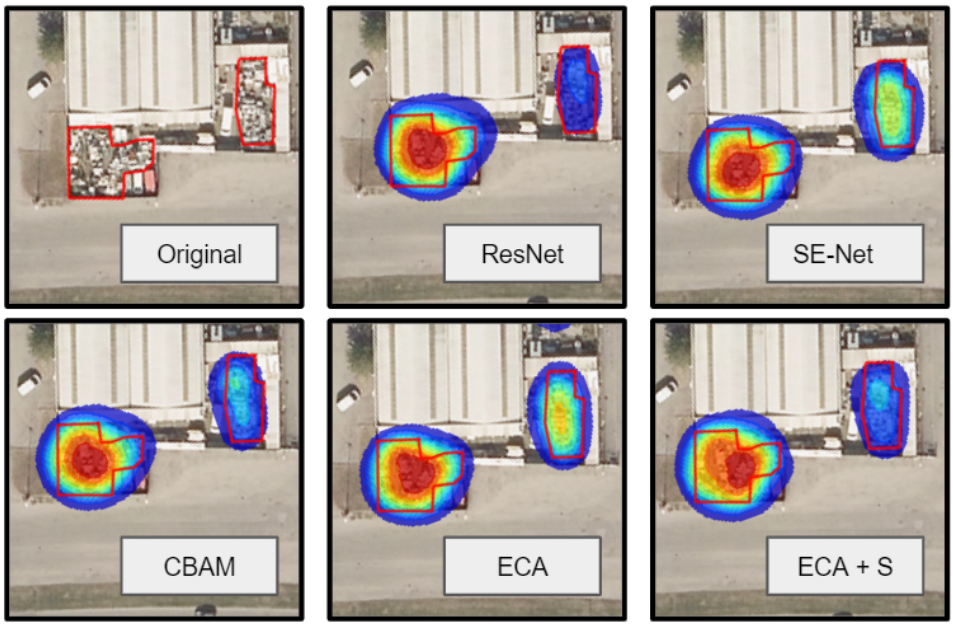


Figure 7.9: Class Activation Maps for all the models, generated at $t = 0.25$, that recognize all the annotated objects.

Moreover, by visually analyzing many examples, we noticed some cases in which all the models' CAMs signaled some region that at first glance seemed irrelevant, since it was not covered by any annotation. Looking closely at the images, those spots had objects that resembled one of the waste types that should have been annotated, which might suggest a distraction by the human annotator and could be added to the dataset. The image in Figure 7.10 is a crop of the image in Figure 7.3, where this kind of behaviour is present. It's clearly visible inside the yellow box a group of scattered wastes that should have been annotated but was missed.

Finally, a difficulty that the CAMs of all the models present is their ability to separate relevant objects that are nearby. This problem arises on many occasions, since in an IL many wastes are collected in a confined space. While a human operator could easily distinguish different shapes or different kinds of wastes that are placed near to each other, for the CAMs it is much

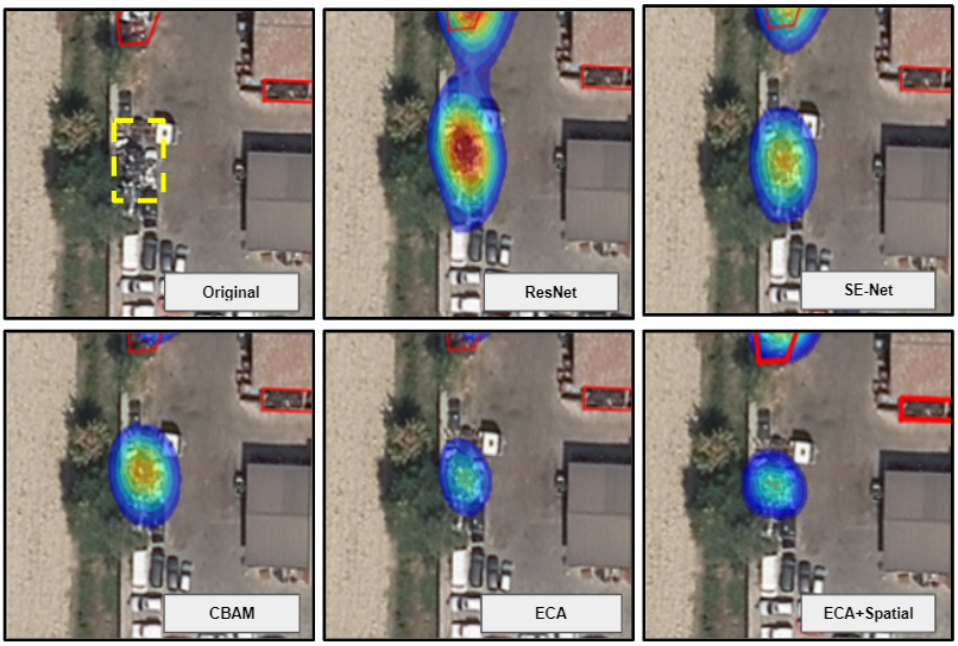


Figure 7.10: Class Activation Maps for all the models, generated at $t = 0.25$. Cropped from Figure 7.3.

more difficult. This could also be a serious problem going forward with the development of the CAMs as a technique to produce good segmentation masks or bounding boxes for the relevant objects in the scene. An example of this behaviour is seen in Figure 7.11, where the 2 groups of scattered wastes on the left and the pallets on the right were separated by the human annotator, while the CAMs recognize the whole but make a unique blob.

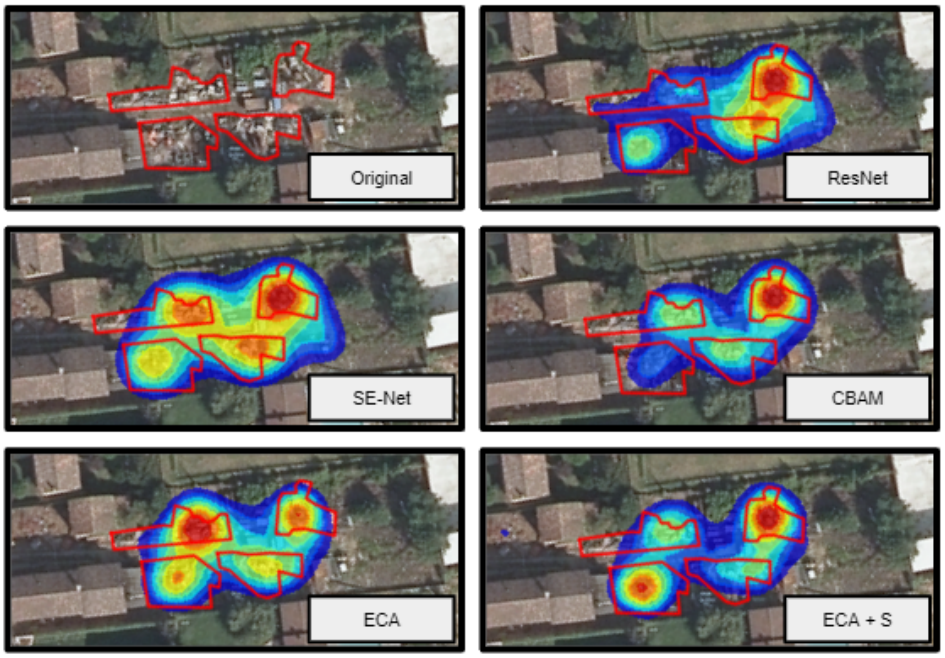


Figure 7.11: Image containing four distinct relevant objects. Class Activation Maps generated at $t = 0.25$.

Chapter 8

Conclusions and Future Works

This work had multiple objectives. Starting from an existing CNN-based classifier, trained on aerial images to predict if the scene under examination contains an illegal landfill, the first step was to study the benefits that could be obtained with the integration of Attention mechanisms.

First of all, the necessity and the challenges related to the task of automating illegal landfills detection were introduced. In order to do this, the state of the art was presented, showing the advancement of the research, starting from the early '80 up until the ongoing efforts. Examining the main objective of this task, that is to enable mass-scale automatic territory monitoring, it has been seen that one of the most recent and promising directions is the use of Deep Learning techniques.

Next, we explained the techniques that were used in this research to enhance the classification capabilities, the Attention mechanisms. Those mechanisms aim is to focus the computational resources on the smallest and most informative parts of the input data. In particular, three Attention modules, applicable as plug-and-play blocks inside pre-existing CNNs architectures, have been described in detail: the Squeeze-and-Excitation [23], the Convolutional Block Attention Module [60], and the Efficient Channel Attention [58]. All three of these modules aim to produce more refined feature maps, exploiting their inter-channel and cross-channel relationship. Squeeze-and-Excitation (SE) is simpler and it is the precursor of the other two. Convolutional Block Attention Module (CBAM) refines the SE module and tries to model also the spatial relationships. Efficient Channel Attention (ECA) instead aims at lowering the complexity of the algorithm while also exploiting the local cross-channel interactions.

Then it was shown how the dataset used for the experiments is composed. Starting from ∼3000 aerial images taken from the Lombardia region, this dataset was used to perform a binary classification task: the positive class being the presence of an illegal landfill in the image, the negative one the absence. This is then followed by the description of all the different configurations that have been experimented with. Starting from the three attention modules taken from the literature, we changed some of their configurations and tried to mix some elements between different modules. In particular, we experimented with a novel configuration based on the ECA channel attention module combined with the spatial attention module of CBAM, called ECA+Spatial. Then, having all the training process lined out, the metrics used to evaluate the resulting trained model were presented. Finally, the results were compared. First, we evaluated the various configuration of the CNNs enhanced with the different attention modules. Then, we compared the original versions of SE, CBAM, and ECA, the ECA+Spatial module that we defined, and the baseline architecture of ResNet. In this comparison, it has been seen that the SE module was the worst method among the attention ones, while the best one was ECA, which also managed to surpass ResNet by a good margin on our metrics of interest. Moreover, the ECA+Spatial module was not able to improve the performances of ECA further. A qualitative analysis has also been conducted, showing with some example images other insights on how the models behave.

The second objective of this thesis was to work on the interpretability of the networks' predictions with the Class Activation Maps (CAMs). Moreover, we were interested in seeing if the implementation of the attention mechanism lead to more accurate CAMs. In order to do so, we made annotations of the relevant objects on a subset of images of the original dataset, obtaining a new dataset with 596 images and 3411 annotations. Using this dataset, the CAMs produced by the models with attention have then been compared to the ones generated with the baseline ResNet. Four main measures have been used in this comparison. The Intersection over Union metric has been used to verify how similar the areas highlighted by the CAMs are with respect to the Ground Truths. With the other metrics, Irrelevant Attention and Annotation Coverage, we checked how many objects were found by the generated CAMs and how much the CAMs were out of focus, looking at the percentage of the CAM areas that was not covering any GT. The results were that the attention-enhanced modules were able to produce better CAMs, proving that with them the models effectively concentrate on the relevant objects. ECA in particular was the best one, with relevant gains on every metric. Moreover, we found that in general the CAMs, given a threshold around ∼0.35 and a coverage threshold of 0.3, were capable of covering around ∼50% of the annotated ground truths, proving that the Attention mechanisms are capable of localizing the complex objects that are present in the aerial scene of an illegal landfill. A final qualitative analysis gives

some additional insights on how the CAMs could behave if used as a tool for object detection.

8.1 Future Work

Starting from the basis of this research, future efforts may be invested following multiple separated lines. In particular, it is proposed to:

- Combine multi-scale approaches with attention mechanisms. Parallel researches focused on the multi-scale detection problem. In them, in order to further improve the performance of the baseline ResNet model on the task of illegal landfills classification, models based on the Feature Pyramid Networks [36] were used. This architecture has been able to obtain a great enhancement of the performances. Further immediate improvement on both these research could be to investigate whether the combination of the Attention Mechanisms with the Feature Pyramid Networks could be beneficial.

- Extend the task to a multi-label classification task. The data set is constantly growing with images coming from different sources and more annotations being made on them. In this research we trained a binary classifier on the IL vs. not IL classes. However we also have the information about the objects, defined in Chapter 4, that are present in the samples, and it could be possible to train a multi-label classifier on those object classes.

- Analyzing different CAMs algorithms. Implementing and comparing different CAMs computation methods that could lead to more refined CAMs, like Grad-CAM [53], Grad-CAM++ [7], and Smooth Grad-CAM++ [46]. In fact, these different algorithms have already proved in numerous other works to lead to more refined CAMs, and having them could in turn be a better starting point for the next development.

- Weakly-supervised object detection. After some further improvements and optimization of the CNN and of its CAMs, it could be possible to exploit the latter, computed from a whole-image classifier, to automatically create the training data set of a weakly supervised detection and segmentation model. It is suggested to study and apply some post-processing techniques, like displacement fields, cluster centroids, and conditional random walks (as illustrated in [3]) to further refine the created CAMs, in order to obtain the high-quality segmentations necessary for training an object detector. This is particularly relevant because it would aid in one of the most time-intensive processes, which is the manual annotation of the datasets.

Bibliography

[1] O. Adedeji and Z. Wang, "Intelligent waste classification system using deep learning convolutional neural network", *Procedia Manufacturing*, vol. 35, pp. 607–612, 2019.

[2] D. Affinito, "Lombardia, i furbetti dei rifiuti ≪pizzicati≫ dal cielo con il satellite spia", *Corriere della Sera: Dataroom di Milena Gabanelli*, Mar. 2021.

[3] J. Ahn, S. Cho, and S. Kwak, "Weakly supervised learning of instance segmentation with inter-pixel relations", *CoRR*, 2019.

[4] G. Biotto, S. Silvestri, L. Gobbo, E. Furlan, S. Valenti, and R. Rosselli, "Gis, multi-criteria and multi-factor spatial analysis for the probability assessment of the existence of illegal landfills", *International Journal of Geographical Information Science*, vol. 23, no. 10, pp. 1233–1244, 2009.

[5] T. Bluche, "Joint line segmentation and transcription for end-to-end handwritten paragraph recognition", in *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[6] C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu, D. Ramanan, and T. S. Huang, "Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks", in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2956–2964.

[7] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks", in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[8] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 6298–6306.

[9] G. Cheng, X. Xie, J. Han, L. Guo, and G.-S. Xia, "Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities", *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 3735–3756, 2020, ISSN: 2151-1535.

[10] *Commissione parlamentare di inchiesta sul ciclo dei rifiuti e sulle attività illecite ad esso connesse*. [Online]. Available: `https://inchieste.camera.it/rifiuti/home.html?multiLeg=true`.

[11] *Communication from the commission to the european parliament, the council, the european economic and social committee and the committee of the regions eu actions to improve environmental compliance and governance*, Jan. 2018.

[12] M. Corbetta and G. Shulman, "Control of goal-directed and stimulus-driven attention in the brain", *Nature reviews. Neuroscience*, vol. 3, pp. 201–15, Apr. 2002.

[13] B. De Carolis, F. Ladogana, and N. Macchiarulo, "Yolo trashnet: Garbage detection in video streams", in *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, IEEE, 2020, pp. 1–7.

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

[15] T. L. Erb, W. R. Philipson, W. L. Teng, and T. Liang, "Analysis of landfills with historic airphotos", *Photogrammetric Engineering and Remote Sensing*, vol. 47, no. 9, pp. 1363–1369, 1981.

[16] European Commission, Directorate-General for Environment, *Environmental compliance assurance - guidance document*, 2019.

[17] D. Garofalo and F. Wobber, "Solid waste and remote sensing", *Photogrammetric engineering*, vol. 40, no. 1, pp. 45–59, 1974.

[18] R. Girshick, "Fast r-cnn", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[19] R. Hataya. (2017). "Se-resnet50 pretrained weights", [Online]. Available: `https://github.com/moskomule/senet.pytorch/releases/download/archive/seresnet50-60a8950a85b2b.pkl`.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[21] ——, *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, 2015. arXiv: `1502.01852`.

[22] D. Hoornweg and P. Bhada-Tata, "What a waste: A global review of solid waste management", *Urban development series, knowledge Paper*, vol. 15, pp. 87–88, Jan. 2012.

[23] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[24] INTERPOL - UN Environment, *Strategic report: Environment, peace and security – a convergence of threats.* 2016.

[25] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, 2015. arXiv: 1502.03167.

[26] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1254–1259, Dec. 1998.

[27] M. Jaderberg, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks", *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, Jun. 2015.

[28] G. Jakovljevic, M. Govedarica, and F. Alvarez-Taboada, "A deep learning model for automatic plastic mapping using unmanned aerial vehicle (uav) data", *Remote Sensing*, vol. 12, no. 9, 2020.

[29] R. Jordá-Borrell, F. Ruiz-Rodriguez, and Á. L. Lucendo-Monedero, "Factor analysis and geographic information system for determining probability areas of presence of illegal landfills", *Ecological Indicators*, vol. 37, pp. 151–160, 2014.

[30] S. Kaza, L. Yao, P. Bhada-Tata, and F. Van Woerden, *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050.* The World Bank, 2018.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12, Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105.

[32] H. Larochelle and G. E. Hinton, "Learning to combine foveal glimpses with a third-order boltzmann machine", in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23, Curran Associates, Inc., 2010.

[33] *Legambiente.* [Online]. Available: https://www.treccani.it/enciclopedia/legambiente.

[34] Legambiente, *Ecomafia 2020. Le storie e i numeri della criminalità ambientale in Italia*, ser. Saggistica ambientale. Edizioni Ambiente, 2020, ISBN: 9788866272878.

[35] A. Limoli, E. Garzia, A. De Pretto, and C. De Muri, "Illegal landfill in italy (eu)—a multidisciplinary approach", *Environmental Forensics*, vol. 20, no. 1, pp. 26–38, 2019.

[36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick, "Microsoft coco: Common objects in context", May 2014.

[38] Y. Liu, Y. Zhong, and Q. Qin, "Scene classification based on multiscale convolutional neural network", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 12, pp. 7109–7121, 2018.

[39] A. L. Lucendo-Monedero, R. Jordá-Borrell, and F. Ruiz-Rodriguez, "Predictive model for areas with illegal landfills using logistic regression", *Journal of Environmental Planning and Management*, vol. 58, no. 7, pp. 1309–1326, 2015.

[40] D. O. Melinte, A.-M. Travediu, and D. N. Dumitriu, "Deep convolutional neural networks object detector for real-time waste identification", *Applied Sciences*, 2020.

[41] A. Miech, I. Laptev, and J. Sivic, *Learnable pooling with context gating for video classification*, 2018. arXiv: 1706.06905.

[42] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997, ISBN: 978-0-07-042807-2.

[43] G. Mittal, K. B. Yagnik, M. Garg, and N. C. Krishnan, "Spotgarbage: Smartphone app to detect garbage using deep learning", in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 940–945.

[44] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines", in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10, Haifa, Israel: Omnipress, 2010, pp. 807–814.

[45] P. Nowakowski and T. Pamula, "Application of deep learning object classifier to improve e-waste collection planning", *Waste Management*, vol. 109, pp. 1–9, 2020.

[46] D. Omeiza, S. Speakman, C. Cintas, and K. Weldemariam, "Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models", *CoRR*, 2019. [Online]. Available: http://arxiv.org/abs/1908.01224.

[47] J. Park. (2018). "Resnet50+cbam pretrained weights", [Online]. Available: https://www.dropbox.com/s/bt6zty02h9ibufi/RESNET50_CBAM_new_name_wrap.pth?dl=0.

[48] S. R., R. P., V. S., K. R., and G. M., "Deep learning based smart garbage classifier for effective waste management", in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 2020, pp. 1086–1089.

[49] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[50] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", *CoRR*, vol. abs/1804.02767, 2018. arXiv: 1804.02767.

[51] R. Rensink, "The dynamic representation of scenes", *Visual Cognition*, vol. 7, pp. 17–42, Jan. 2000.

[52] F. Rosenblatt, "The perceptron - a perceiving and recognizing automaton", Cornell Aeronautical Laboratory, Ithaca, New York, Tech. Rep. 85-460-1, Jan. 1957.

[53] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization", in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[54] S. Silvestri and M. Omri, "A method for the remote sensing identification of uncontrolled landfills: Formulation and validation", *International Journal of Remote Sensing*, vol. 29, no. 4, pp. 975–989, 2008.

[55] W. Tong, W. Chen, W. Han, X. Li, and L. Wang, "Channel-attention-based densenet network for remote sensing image scene classification", *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 4121–4132, 2020.

[56] L. Torrey and J. Shavlik, "Transfer learning", in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, IGI global, 2010.

[57] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[58] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks", in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 531–11 539.

[59] Y. Wang and X. Zhang, "Autonomous garbage detection for intelligent urban management", in *MATEC Web of Conferences*, EDP Sciences, vol. 232, 2018, p. 01 056.

[60] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module", in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[61] B. Wu. (2019). "Eca-net50 pretrained weights", [Online]. Available: `https://drive.google.com/open?id=1670rce333c_lyMWFzBlNZoVUvtxbCF_U`.

[62] G.-S. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, L. Zhang, and X. Lu, "Aid: A benchmark data set for performance evaluation of aerial scene classification", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3965–3981, 2017.

[63] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention", in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, PMLR, Jul. 2015, pp. 2048–2057.

[64] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification", in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, 2010, pp. 270–279.

[65] K. Yun, Y. Kwon, S. Oh, J. Moon, and J. Park, "Vision-based garbage dumping action detection for real-world surveillance platform", *ETRI Journal*, vol. 41, no. 4, pp. 494–505, 2019.

[66] X. Zhao, J. Zhang, J. Tian, L. Zhuo, and J. Zhang, "Residual dense network based on channel-spatial attention for the scene classification of a high-resolution remote sensing image", *Remote. Sens.*, vol. 12, p. 1887, 2020.

[67] X. Zhao, J. Zhang, J. Tian, L. Zhuo, and J. Zhang, "Residual dense network based on channel-spatial attention for the scene classification of a high-resolution remote sensing image", *Remote Sensing*, vol. 12, no. 11, 2020.

[68] Z. Zhao, J. Li, Z. Luo, J. Li, and C. Chen, "Remote sensing image scene classification based on an enhanced attention module", *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020.

[69] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization", in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2921–2929.