LUISS

Degree Program in Data Science and Management

Course of Machine Learning

Exploring Explainable AI (XAI) in Computer Vision: A Practical Application to Satellite Image Scene Classification

Prof.Giuseppe Francesco Italiano

SUPERVISOR

Prof. Blerina Sinaimeri

CO-SUPERVISOR

760721

CANDIDATE

Academic Year 2023/2024

Table of Contents

1. Introduction

1.1 Background of the study

Nowadays trend clearly highlights that almost all industries, which have embraced information technologies in recent years, have their foundation in Artificial Intelligence (AI) models and applications. Despite the fact that AI is not exactly a novel topic of research, tracing its roots almost a century ago, it is evident that it has been experimenting in this decades an unprecedent popularity and diffusion, permeating not only emerging tech industries, such as fintech or insurtech[1], but also traditional ones. The most exemplificative case is for sure the healthcare sector, whose potentiality have been unleashed by extensive integration of Machine Learning (ML) and AI models together with human preexisting knowledge. The root cause behind this incredible phenomenon relies in the intrinsic quality of ML/AI, which by definition are able to learn, reason and adapt, and therefore can tackle ever-more-complex tasks, and be deployed to most of use cases. At the beginning of this steep learning curve, AI was solving very simple tasks relying to extremely frugal architecture, but lately, AI-powered systems have become so sophisticated that very little human involvement is needed in their development or implementation. When choices made by these kinds of systems eventually impact the lives of people, there is a growing requirement to comprehend how AI techniques provide these decisions. The risk is in making and applying decisions that are illegitimate, or that just make it impossible to get thorough justifications for their actions. In precision medicine, for instance, specialists need significantly more information from the model than a simple binary prediction to support their diagnosis, therefore explanations that back up the model's output are essential. This is just an example, but comparable scenarios could be found in almost all field where AI-systems are deployed. This emerging need of understanding culminated in what today is called Explainable Artificial Intelligence (XAI).

The diffusion of XAI is driven by several key factors. Regulatory and ethical considerations play a crucial role. In fact, there are always increasing regulatory requirements for transparency and accountability in AI systems. The main regulatory policy in Europe, which mandates explainability to ensure non-discriminatory practices, is the GDPR[2]. Ethical concerns over fairness, bias, and accountability also drive the demand for ethical AI to build trust among users and stakeholders. Trust and transparency are essential to provide clear and understandable explanations, necessary to reduce

[1] Investopedia define “insurtech” as the use of technology innovations designed to find cost savings and efficiency from the current insurance industry model

[2] General Data Protection Regulation, is a European Union regulation on information privacy in the European Union (source https://en.wikipedia.org)

the "black box" nature of AI, particularly in critical domains like healthcare, finance, and legal systems. Industry adoption and business value are significant drivers, with companies recognizing the competitive advantage of using XAI to improve decision-making processes, enhance customer satisfaction, and reduce risks associated with AI deployment. Technological advancements have also contributed to XAI diffusion, thanks to developments in techniques and tools for making AI models interpretable. Stakeholder demand and public pressure, including consumer demand for transparency, further drive the need for XAI. By addressing these drivers, organizations can ensure AI systems are trustworthy, ethical, and aligned with regulatory and societal expectations.

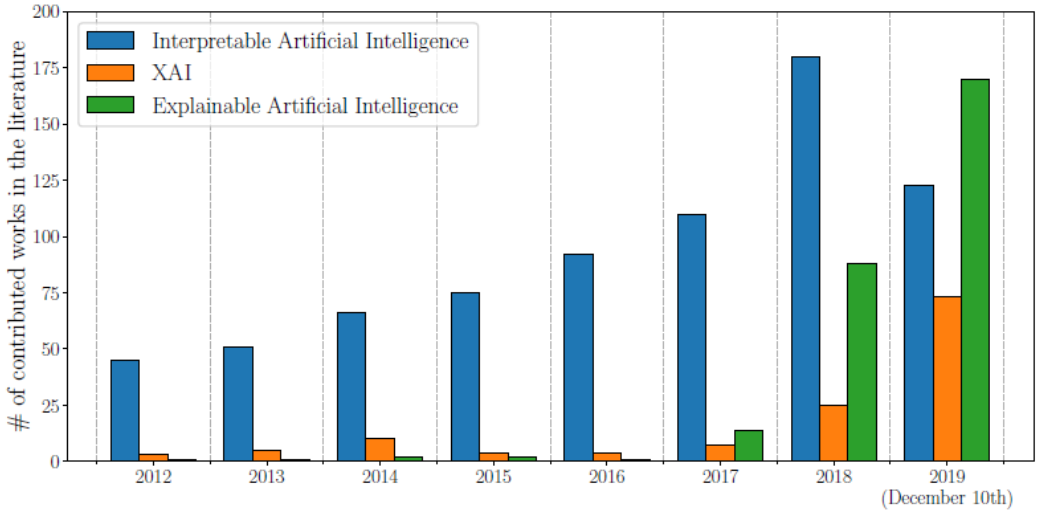

Figure 1: Evolution of the number of total publications whose title, abstract and/or keywords refer to the field of XAI, source: "Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI" (Alejandro Barredo Arrieta, 2020)

All those factors, as showed in Figure 1, led to an incredible exponential increase in literature interest and publications within the realm of interpretable and explainable AI.

Despite significant advancements, the literature on Explainable Artificial Intelligence (XAI) reveals several persistent gaps. There is a lack of standardization regarding what constitutes effective explanations and the metrics to evaluate them, compounded by insufficient human-centric evaluation that overlooks user interaction and cognitive interpretation. Scalability remains a challenge, with current methods often being computationally intensive and impractical for real-time applications, especially for complex models. Explanations frequently lack context-sensitivity and domain-specific relevance, and there is limited interdisciplinary collaboration with fields like psychology and human-computer interaction. The trade-off between model transparency and performance, alongside issues of bias and fairness in explanations, further complicates the landscape. Furthermore, these explanation needs to be interpreted. While for domain-experts it could be straightforward, common

users should be aided and guided through this process. This poses the greatest gap in the existent literature because all the effort to “explain” AI systems is completely meaningless if the counterpart is not provided with sufficient tools to meet this effort.

Addressing these gaps requires an interdisciplinary approach to develop user-centric, efficient, and context-aware explanation methods.

1.2 Research Objectives

The primary research objective of this thesis is to provide a comprehensive guide about XAI importance and diffusion in recent years, with a specific focus on CNN-tailored techniques. It would be described in details not only their functioning, but also how to effectively interpret them, aiming to reduce the literature gap in that sense. This would be the greatest contribution of this work. To reach a complete understanding, it is necessary a glance view of both theoretical foundations and practical output, focusing on which problem is addressed and how. In order to reach this goal, in this dissertation will be developed and evaluated an explainable artificial intelligence (XAI) model for image classification of satellite images. Results chapter will show that enhancing understandability does not necessarily imply a decrease in overall accuracy and effectiveness. Therefore, secondary research goals are:

- Perform XAI techniques to a CNN Model: To implement state-of-art CNNs models for satellite image classification, and applying different techniques to make it clear and interpretable.
- Evaluate Model Performance: To assess the accuracy and robustness, both in terms of accuracy and interpretability.
- Interpretability Analysis: To analyze and quantify the understandability of the explanations, ensuring they are understandable to domain experts and non-experts alike.

Thesis hypothesis, indeed, can be summarized as:

- Accuracy-Interpretability Tradeoff: This hypothesis is based on the premise that interpretability does not necessarily compromise model performance, showing at the same time that also “black box” can be fully explained.
- Interpretability Hypothesis: This hypothesis assumes that the designed XAI techniques will effectively bridge the gap between model complexity and user understanding, prosing understandability as standard means of evaluation of each ML/XAI models.

- Two-folded XAI benefits: This last hypothesis states that XAI techniques are beneficial not only to external users, but also to model designers, contributing to reduce bias in data and to diagnose problems.

1.3 Structure of the Thesis

This thesis is structured in 3 main chapters: Literature Review, Practical Applications and Conclusions.

Literature Review chapter deep dives into state-of-art of three main topics: Explainable AI, Convolutional Neural Networks and Explainable AI applied to Neural Networks. The first subchapter, about XAI, starts from the definition of XAI, analyzing then the main drivers for XAI implementation according to the literature. It follows a detailed description of XAI taxonomy, with a specific focus about XAI "units of measure". The last part explains models' classification in terms of explainability, and XAI techniques classification, according to target model and means of explanation.

CNN literature review, indeed, begins with a brief summary about Artificial Neural Networks background works and their standard form definition. Then, it continues with CNN architecture description, with major focus on its distinctive elements, such as convolutional layers and pooling. The section ends with the analysis of fundamental steps and papers which lead to CNNs wide diffusion and to state-of-art implementations. The first chapter's last section delves into most popular XAI techniques tailored for CNNs, describing for each of them their functioning, their first introduction and all the further modifications and adaptations.

The following chapter, which constitutes the main part of the whole thesis, regards the practical application. Firstly, it is described the satellite images classification tasks, deepening factors such as its practical importance, i.e., social impact, and literature review about ML/AI models deployed during the years. Next section, indeed, enumerates all viable datasets provided by the literature, together with a detailed explanation that led to final dataset choice. After this, the next subparagraph analyzes selected models' structure, focusing on both theorical background and final architecture. Before the result section, it is explained in details how to interpret the output of each XAI technique deployed. This is one of the greatest contributions of this work, since the literature lacks of an exhaustive review about it. Finally, two models' performances are compared according traditional techniques, such as accuracy metrics and confusion matrixes, and, most importantly, through interpretation of XAI techniques visual outputs, with tackle attributes like generalization ability, feature relevance and correctness of decision process.

Lastly, the conclusion chapter summarizes thesis findings and results, both practical and theoretical, and highlights the major contribution together with research's limitations. The whole dissertation is concluded by recommendations about future works and hypothesis about future research directions.

2. Literature Review

2.1 XAI

Artificial intelligence (AI) dimension has recently shifted from a technological niche to a global innovation driver, penetrating very diverse sectors. Among them, it is possible to highlight healthcare, finance, transportation, and education. As AI continues to evolve, its potential to drive economic growth, enhance human capabilities, and address complex societal challenges underscores its pivotal role in shaping the future. The extremely dangerous phenomenon, which characterized all major innovations, is the tendency to abuse of AI, taking advantage of the scarcity of knowledge of the generalist public (i.e., common users). Within this framework the need to explain or interpret, depending on the point of view, arise. Explainable Artificial Intelligence (XAI) refers to methods and techniques in the field of AI that provide human-understandable explanations of how models make their decisions. XAI aims to make the output of AI systems more transparent, understandable, and trustworthy, particularly for complex models like deep neural networks, whose operations can otherwise be opaque and difficult to interpret. This is achieved through various tools and approaches that elucidate the decision-making processes. There are three main "design driver" (Alejandro Barredo Arrieta, 2020) in ensuring interpretability while building a ML model:

- "Interpretability helps ensure impartiality in decision-making"; the most common example in that sense regards biased training data.
- "Interpretability facilitates the provision of robustness by highlighting potential adversarial perturbations that could change the prediction"; a practical example can be found in image classification. If, for instance, changes in medical images background influences the prediction, some adjustments must be implemented for both training data and process.
- "Interpretability can act as an insurance that only meaningful variables infer the output"; which it is translated in deep diving in causal logical relations between covariates and output.

On the other hand, XAI involves not only domain experts, but also common users and regulatory entities. Literature so far delineated XAI goals which affects specifically these two target groups as:

- Fairness: refers to the ethical and methodological principles and practices that seek to ensure that AI systems operate impartially, justly, and equitably. The concept encompasses a variety of specific measures and considerations designed to prevent discrimination and bias that can be inherent in AI algorithms, particularly those that affect decisions impacting humans.
- Trustworthiness: refers to the degree of confidence in the model to act as expected.

- Privacy awareness: refers to the recognition and proactive management of privacy risks and concerns associated with the collection, storage, processing, and sharing of personal data by AI systems.

After enumerating all main factors which makes this dissertation actual and meaningful, the next paragraph will investigate further what is commonly intended as "explainability", how this concept evolved during time, and how to achieve it in the realm of Machine Learning and Artificial Intelligence.

The first important distinction to be made, within the scope of XAI taxonomy, regards two foundational terms in this field: interpretability and explainability. Interpretability is a "passive characteristic" of a ML model, and it is referred to the degree up to which a human being can makes sense of it. A synonym for this specific term is transparency. On the other hand, explainability express an "active characteristic", describing the proactive effort of a model to clarify internal processes and final output (Alejandro Barredo Arrieta, 2020). In this landscape, the term "understandability" represents a perfect synthesis between interpretability and explainability. The point where the explainability effort meets the interpretation process is the degree of understanding of the user. Hence, understandability is the measure of effectiveness for each XAI technique.

Within the scope of this dissertation, a XAI technique is defined as the set of practice, transformation and modification applied to a specific ML or AI model in order to make it understandable, from both designer and user perspective. In the literature, there can be highlighted two models' macro-categories: model which are interpretable due to their design, and those which needs external methods (i.e., XAI techniques) in order to be explained. Hence, in the literature, the first category is also known as "interpretable models", while the second one is called "post-hoc" explainable models (Riccardo Guidotti, 2019). The same papers (Riccardo Guidotti, 2019) proposed three levels of transparency among interpretable models, consisting in:

- "algorithmic transparency", occurring when a user is able to follow and understand the process pursued by the model.
- "decomposability", standing for the ability to explain each distinct model's parts.
- "simulatability", referring to the capacity of a system or process to be simulated by a human.

Among "transparent box" models, there are: Linear/Logistic Regression, Decision Trees, K-Nearest-Neighbors, General Additive Models, Rule-Based Learners and Bayesian Models. Analyzing, for instance, simple Linear Regression models, it is possible to denote that it has high "simulatability", since predictors are human readable and their interaction is limited. "Decomposability", instead,

depends on the number of predictors and the degree of their interactions. Lastly, “algorithmic transparency” can be deemed as very low, due to the computational complexity which requires mathematical tools and background. On the other hand, decision trees constitute an optimal choice in terms of transparency. “Simulatability” is very high, since every user can replicate tree prediction without need of complex tools. “Decomposability” is high too, due to the fact that model’s rules preserve data readability. In the end, decision trees decision-making process has highly human comprehensible features, granting direct understanding and therefore high “algorithmic transparency”.

If a model fails in meeting above mentioned transparency levels, some methods must be applied, in order to reach XAI. In this case the system can be considered as a “black-box”. For such models, literature proposes different distinctions and categorizations. Most widely used was introduced in 2017 (Doran, 2017), suggesting a tripartition into: “opaque systems”, where connection between input and output is invisible to users; “interpretable system”, where through mathematical analysis it is possible to understand input-output mapping; “comprehensible systems”, in which the model return not only the output, but also “symbol and rules”, enhancing human readability. Since most ML/AI techniques lies in one of these three categories, a wide set of “post-hoc” interpretability techniques arose through the years. These can be divided according to two dimensions: the model to which these techniques can be applied to, and the means through which they enhance understandability. For what concerns the first dimension, it is possible to perform “agnostic techniques”, which refers to set of techniques which can be applied to all models without any regards to type of task, data, internal process or final output; and “model specific techniques”, which are all these techniques designed and tailored for different ML models. Among this last category, it is possible to perform a more specific partition between techniques applicable to “shallow models”, and ones designed for “deep learning” models. A practical example of an agnostic XAI technique is SHAP (SHapley Additive exPlanations). SHAP is a model-agnostic method that uses game theory, specifically Shapley values, to explain the output of any machine learning model. For what concerns shallow models, a notable example of a model-XAI technique is the use of single Decision Tree Visualizations in ensemble learnings. Decision trees are inherently interpretable due to their structure, which resembles a set of decision rules. In deep learning realm, instead, a well-known example is the use of Attention Mechanisms in neural networks, particularly in models like transformers used for natural language processing (NLP). Attention mechanisms allow the model to focus on specific parts of the input data that are most relevant for making a prediction. In NLP, this means highlighting certain words or phrases that are crucial for understanding the context or meaning of a sentence or document.

On the other hand, considering classification method based on techniques means, there are:

- "Text explanations" which address the issue of making a model more explainable by developing the ability to produce written explanations. Text explanations also cover each process for creating symbols that indicate how the model operates. An example can be Rule Extraction from a Support Vector Machine (SVM). In fact, after making a diagnosis, the model can generate a textual explanation by extracting decision rules that influenced the outcome.
- "Visual explanation" techniques that try to depict the behavior of the model visually. Numerous visualization approaches found in the literature are combined with dimensionality reduction strategies to provide basic visualizations that are easy for humans to understand. Visualizations are thought to be the best method for introducing complicated interactions among the variables in the model to users who are unfamiliar with machine learning modeling. Visual explanation for a random forest model, for instance, can be Feature Importance Plot.
- "Local explanations" which address explainability dividing the solution space into smaller subspaces, thus providing appropriate explanations for the less complex scenarios. A suitable use case is using Partial Dependence Plots (PDPs) to provide local explanations for individual predictions.
- "Explanations by example" which examine how data examples related to a particular model's output might be extracted to help better understand the model itself. They primarily focus on identifying representative examples that capture the inner relationships and correlations discovered by the model. In a KNN application, an explanation by example can be given by showing the nearest neighbors that led to a classification.
- "Explanations by simplification" that denote techniques in which a new system is built based on the original model. This new, more straightforward approach often aims to minimize complexity while maintaining a comparable performance score. A straightforward use case can be Linear Model Approximation of a Gradient Boosting Machine. Obviously, this simplified model fails in capturing all the nuances of Gradient Boosting but it is useful to give stakeholders an intuitive understanding of how changes in attributes generally affect the output.
- "Feature relevance explanation" methods that calculate a relevance score for each of its variables. These scores express how much a feature modification impacts output. The weight that the model assigns to each of these variables in generating its output can be seen by

comparing the scores of the various variables. Aforementioned SHAP values are a perfect example of feature importance explainer.

To sum up, navigating through the wide sea of XAI taxonomy, the literature agrees to define "understandability" as the ultimate performance metric for XAI techniques. Once this is established, the first distinction to be made concerns models intrinsic "understandability", which lead to a partition between "transparent models" and "black box models". The first ones are characterized by three levels of transparency; "simulatability", "algorithmic transparency" and "decomposability". "Black box" methods, instead, can be divided in "opaque systems", "interpretable systems" and "comprehensible systems".

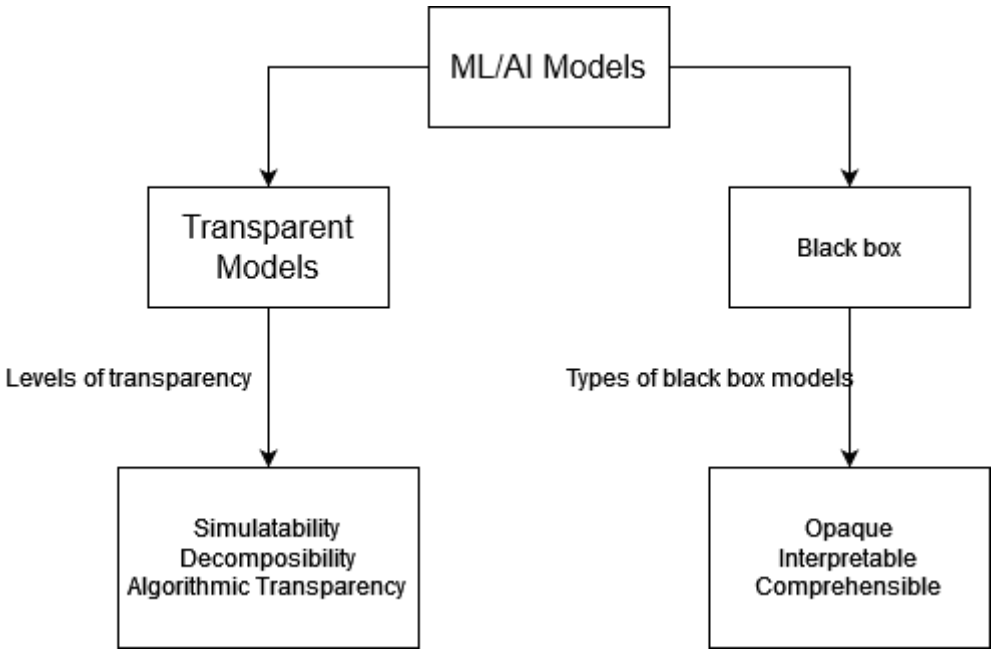

Figure 2: Models Division

Finally, XAI techniques can be categorized by target models ("agnostic" or "specific" techniques), or by means of explanations, which can be "textual", "visual", "local", "by example", "by simplification" and "by feature relevance". The diagram below gives a visual representation of that.

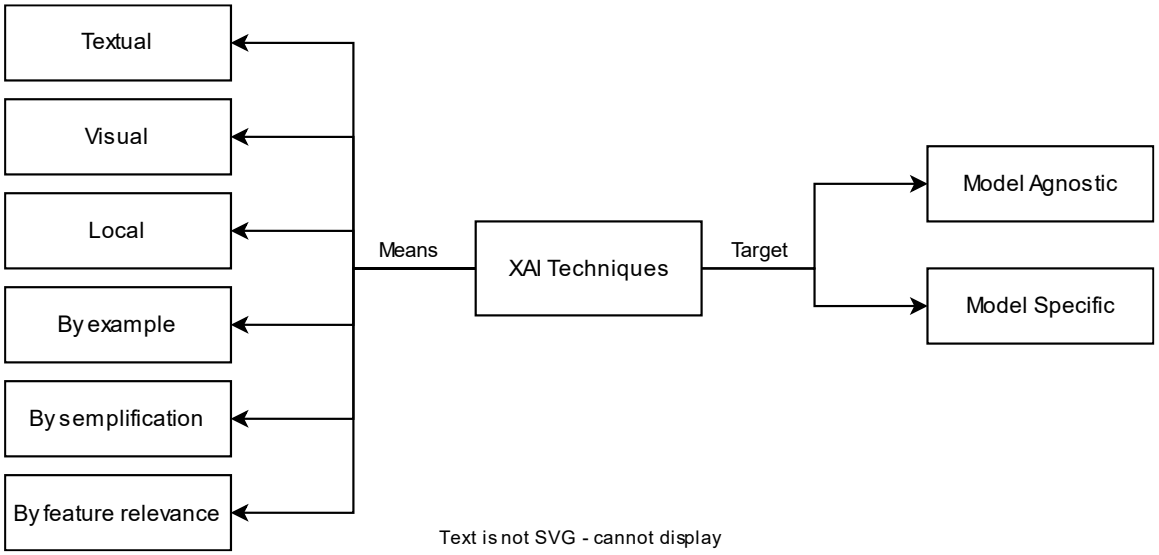

Figure 3: XAI Techniques Divisions

2.2 CNNs

2.2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are one of the main categories within the field of machine learning. The essential idea of ANNs is to mimic human brain functions to solve different types of complex problems. They were introduced for the first time in 1943, in the foundational paper called "*A logical calculus of the ideas immanent in nervous activity*" by McCulloch and Pitts. It is important to recall that this paper authors were "from the University of Illinois, College of Medicine, Department of Psychiatry at the Illinois Neuropsychiatric Institute and the University of Chicago" (McCulloch, 1943), in order to highlight scientific and medical background of this study. The starting point of this paper is that "neural events and the relations among them can be treated by means of propositional logic" (McCulloch, 1943). Then, the authors prosed a computational model for neural networks based on algorithms, which provided the groundwork for what would later evolve into modern artificial intelligence. According to Schmidhuber (2015), "a standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations. Input neurons get activated through sensors perceiving the environment, other neurons get activated through weighted connections from previously active neurons." (Schmidhuber, 2015)

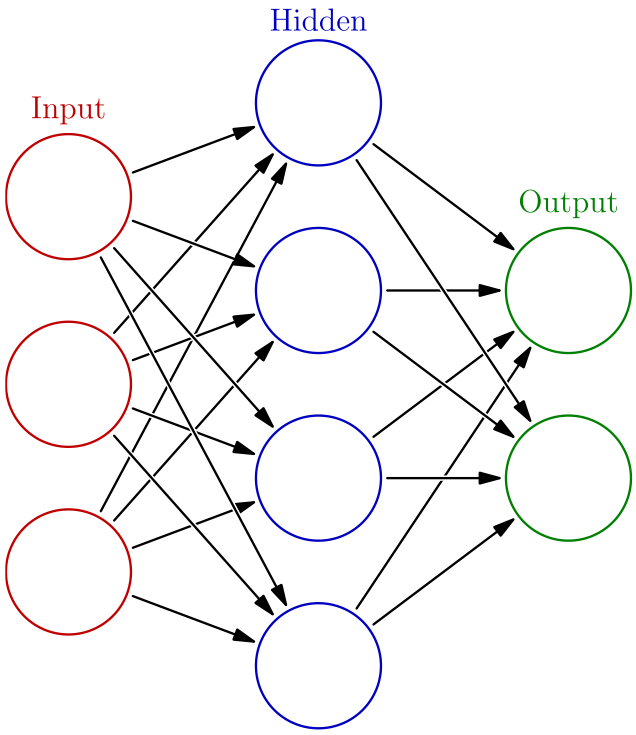

Figure 4: Basic architecture of ANN

The basic architecture of ANNs is showed in Figure 4. The first layer is the so-called input layer, which receives input data in a form that the network can process, i.e., in the form of a vector. Each node in this layer represents a feature of the input data. After the first layer, information is passed to one or more layers, called hidden layers. These are where the actual processing happens. Each neuron receives data from the previous layer, where each input has an associated weight. The highest the weight, the highest the importance. These inputs are multiplied by their respective weights and summed together, often with a constant term added, called bias. This value is then passed through an activation function, which determines the output of the neuron. Common activation functions include sigmoid, tanh, and ReLU (Rectified Linear Unit). These functions add non-linearity to the processing, allowing the network to learn more complex patterns. The final layer is the output layer, which produces the results. The output can be a single value or a vector of values, depending on the task.

Neural nets with few layers, i.e., shallow NN, have being used and discussed from almost a century, while more complex models containing concatenation of nonlinear layers were introduced in the 60s. A cornerstone of ANNs development is without any doubt the introduction of back-propagation technique, in late 70s, and its application to ANNs, which happened for the first time in 1981. In fact, the seminal paper "*Generalization of Backpropagation with Application to a Recurrent Gas Market Model*" (Werbos, 1981), sparked the debate about ANNs, which resulted in the publication of other

papers like "*Learning internal representations by error propagation*" (D. E. Rumelhart, 1986), that "contributed to the polarization of BP for NNs" (Schmidhuber, 2015). Despite the solid theoretical background, some practical limitations delayed the diffusion of ANNs, especially deeper ones. Indeed, it is only fifteen years later, with the start of the new millennium that "deep NNs have finally attracted wide-spread attention, mainly by outperforming alternative machine learning methods such as kernel machines in numerous important applications" (Schmidhuber, 2015). Since this diffusion, lots of modification came up, such as FeedForward Neural Network (FNN), Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN).

2.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are an adaptation of ANNs to image classification tasks. First of all, image classification tasks can be defined as the set of operations aimed to categorize one image, or part of it, into a set of predefined classes (supervised learnings). These tasks can vary from the simple classification to more advanced tasks as object detection, tracking or segmentation. In this framework, CNNs were deployed due to some structural limitations of ANNs. In fact, "one of the largest limitations of traditional forms of ANN is that they tend to struggle with the computational complexity required to compute image data" (O'shea, 2015). Indeed, while ANNs can achieve good performance on small input size data (i.e., MNIST digits dataset[3]), they are not suited to deal with more standard image shape, like for example colored image input of 64x64.

Despite several slight or substantial variation, CNNs, generally, consists of convolutional and pooling layers, grouped into modules, followed, as in ANNs, by fully connected layers. These groups of layers, in most architectures, are stacked on top of each other, forming deep learning models. An example of standard CNN is showed in Figure 5.

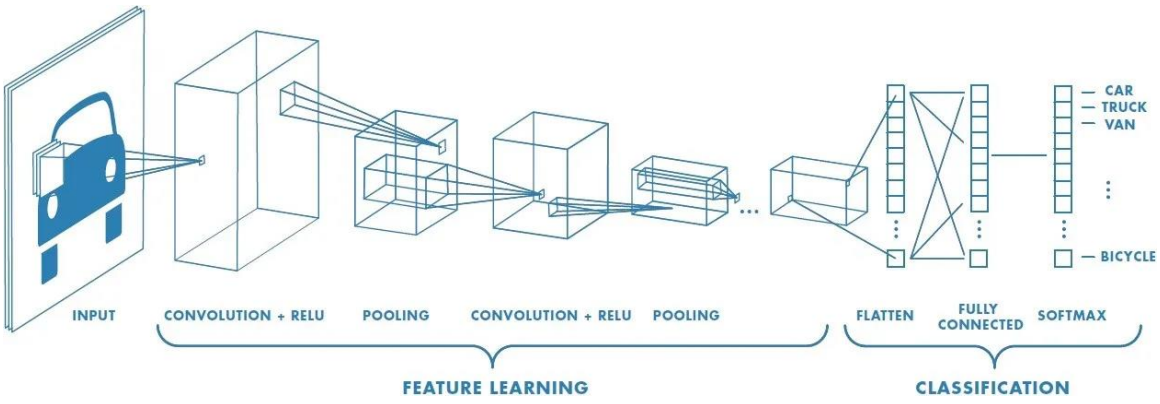

Figure 5: Example of Standard CNN, source: https://www.rolandvarriale.it/i-convolutional-neural-network-cnn/

[3] Large dataset of handwritten digits.

Convolutional layers' function is to extract image features and, consequently, to learn how these features are represented in the input images. Nodes in a convolutional layer are organized into feature maps, and each of them is connected to a set of adjacent nodes from the previous layer, called filter banks (LeCun, 2015). All neurons within a feature map have weights that are constrained to be equal; however, different feature maps within the same convolutional layer have different weights so that several features can be extracted at each location (Yann LeCun, 1998; LeCun, 2015).

On the other hand, pooling layers have a subsampling function. Specifically, their role is to reduce feature maps spatial resolution, in order to restore dimensionality altered by distortions or translations. In order to perform this task, several techniques can be adopted. In early deployed CNNs, the most widely diffused technique was average pooling. It consists in propagating the average input of a circumscribed portion of feature map. More recent models, (Dan C. Ciresan, 2011; Alex Krizhevsky, 2012; Simonyan, 2014; Zeiler, 2014; Szegedy, 2015; Xu, 2015), tend to propagate the greatest value instead of the average one (max pooling). This shift was driven by the possibility, using average pooling, that low activation area could mitigate the presence of high activations regions. Despite the fact that max pooling obtained brilliant empirical results, it has still some limitations. The greatest one consists in the tendency to overfit the model, resulting in a consequent incapability of generalizing on test data (Zeiler, 2014 ; Sainath, Mohamed, Kingsbury, & Ramabhadran, 2013).

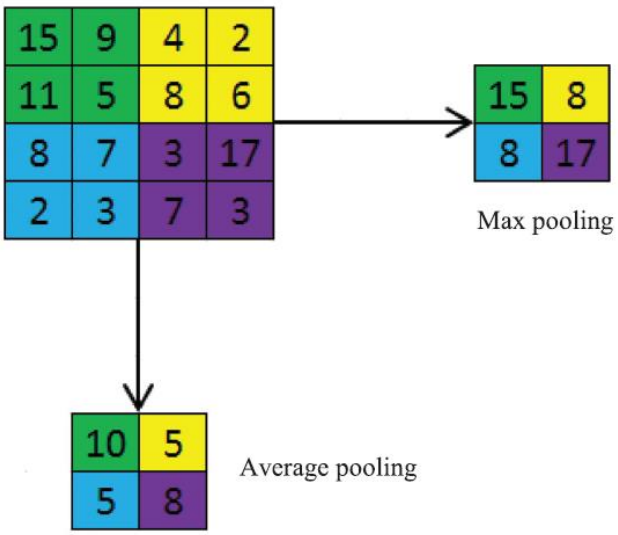

Figure 6: Average versus Max Pooling

source: https://www.researchgate.net/figure/Average-versus-max-pooling_fig1_317496930

These concerns, in recent years, led to development and implementation of new pooling techniques. The firs alternative is provided by L_p pooling, inspired by biological image processing and introduced in the first time in 2009 (Kavukcuoglu, Ranzato, Fergus, & LeCun, 2009), and applied to deep CNNs in 2012 (Sermanet, Chintala, & LeCun, 2012), achieving outstanding results. Among the remarkable innovations within pooling layers framework, it is worth to mention stochastic max pooling (Matthew D. Zeiler, 2013), fractional max pooling (Graham, 2014), mixed pooling (Yu, 2014), and spectral pooling (Rippel, Snoek, & Adams, 2015).

Finally, the last main component of CNNs architecture are fully connected layers. Their role is to mediate between convolutional/ pooling layers and the final output. Their duty is to interpret feature representations coming from convolutional and pooling component, performing though "high-level" reasoning (Sutskever, Hinton, & E, 2012). As showed in Figure 5, the standard operator, applied at the end of the CNNs for classification tasks, is softmax. The softmax operator is a mathematical function that converts a vector of real numbers into probabilities, by exponentiating and then normalizing each element of the vector. It ensures that the resulting probabilities sum to one, which is useful in classification tasks for representing categorical distributions. Also in this field, there is room for discussion and debate. In fact, softmax operator replaced radial basis functions (RBFs), as the classifier on top of the convolutional networks (Yann LeCun, 1998). Latest researches, instead, paved the way for new operators. For example, it was demonstrated an empirically improved accuracy by substituting softmax with Support Vector Machine (Tang, 2013). The paper "*Network in Network*", instead, in order to overcome computational expense issue, suggested the implementation of a global average pooling layer then fed to a linear classifier (Min Lin, 2013). Despite all, "comparing the performance of different classifiers on top of DCNNs still requires further investigation and thus makes for an interesting research direction" (Rawat & Wang, 2017).

The "general" architecture showed and investigated before is the result of the decades of researches, publications and debates. This journey, such as the one of ANNs, start from a neurobiological experiment. In fact, Hubel and Wiesel, discovered that "neurons in the early stages of the primary visual cortex responded strongly to precisely oriented patterns of light, such as bars, but ignored more complex patterns of the input stimulus that resulted in strong responses from neurons in later stages" (Wiesel. & Hubel, 1959). This intuition provided theoretical basis for all CNNs modeled and proposed in the following years.

The second milestone, indeed, is the "neocognitron" model (Fukushima, 1980). This multilayered neural network was modeled around Hubel and Wiesel intuition, mimicking the behavior of both simple and complex cells, resulting in a neural network capable of recognizing simple patterns.

“Neocognitron” paved the way for all CNNs, since that they were derived from it and have a similar architecture (LeCun, 2015).

The third essential step consists in the proposition of the first multilayered CNNs, applied to handwritten zip codes (Yann LeCun, 1998). This model was inspired, as aforementioned, by Fukushima’s work, with one major difference in the training process. In fact, these large-scale neural networks were trained using backpropagation, that, once again, represents a crucial innovation. Despite this turning point, neural network researches diminished between the end of the 90s and the beginning of the new millennium. This is because all kinds of neural networks were considered too hard to train, including CNNs, despite the fact that these ones could rely on faster training compare to standard ANNs. After this transition period, in the second half of 2000s it was possible to observe the phenomenon renamed as “Deep Learning Renaissance”, which obviously involved also the CNNs. The key factors which fueled this “renaissance” were:

- Availability of Large Datasets: The release of large annotated datasets like ImageNet, which contains millions of labeled images across thousands of categories, provided the extensive data necessary for training deep CNNs. This availability of big data allowed neural networks to learn more complex features at multiple levels of abstraction (Olga Russakovsky, 2015).
- Advances in Hardware: Significant advancements in GPU (Graphics Processing Unit) technology enabled the training of deep neural networks much faster than was previously possible with CPUs. GPUs, with their highly parallel architecture, are particularly well-suited to the matrix and vector computations required for deep learning. This reduction in training time opened up new possibilities for experimenting with more complex neural network architectures (Raina, 2009).
- Improved Neural Network Techniques: Innovations in deep learning methodologies, including new activation functions like the ReLU (Rectified Linear Unit), better initialization methods, and effective regularization strategies like dropout, significantly improved the training processes and performance of neural networks. These techniques helped overcome problems such as vanishing gradients and overfitting, which had plagued earlier neural network models (Alex Krizhevsky, 2012).
- Increased Research and Collaboration: There was a marked increase in academic and industrial research focused on deep learning. Collaborative efforts and open competitions, such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), pushed the boundaries of what was possible and demonstrated the superior capabilities of CNNs over

traditional machine learning methods for tasks involving image recognition (Goodfellow Ian, Bengio, & Courville, 2016).

In conclusion, in the current landscape of deep learning, CNNs continue to stand at the edge of innovation, increasing current accuracy and efficiency in image and video analysis. The state-of-the-art CNN architectures now incorporate advanced techniques such as attention mechanisms, which allow models to focus on salient parts of the input data (Ashish Vaswani, 2017), and residual learning, which facilitates the training of exceptionally deep networks by addressing the vanishing gradient problem (Kaiming He, 2016). These innovations have led to the development of highly sophisticated models like EfficientNet and Vision Transformers, which combine the strengths of CNNs and self-attention architectures to achieve remarkable performance across a variety of tasks (Mingxing Tan, 2019; Alexey Dosovitskiy, 2020). Furthermore, the integration of CNNs with other forms of artificial intelligence, such as reinforcement learning and generative adversarial networks, is opening new avenues for applications in autonomous systems, medical image analysis, and multimedia generation (Arun Nair, 2015; Ian Goodfellow, 2014). As the field progresses, ongoing research continues to refine these models, making them not only more powerful but also more accessible and efficient, ensuring that CNNs remain a cornerstone technology in AI for the foreseeable future.

2.3 XAI in CNNs

Since Convolutional Neural Networks fall without any doubts into the "black box" models' category, they are below the ideal understandability landscape. Therefore, CNNs implementations have been study subject for wide set of innovative XAI techniques. For what concerns model agnostic techniques, only LIME technique would be investigated. With regards to model specific techniques, instead, the literature is more structured. According to Ibrahim et al. 2023, "studies related to explaining CNNs can be categorized as decision models and architecture models" (Ibrahim, 2023). Architecture models focus on the network structure, analyzing layers and neurons mechanisms. Among them, it is possible to distinguish between architecture modification models and architecture simplification models. Architecture modification models rely on alteration to CNNs architecture to improve their interpretability. Such changes can imply replacing some components (i.e., layers and loss functions) or adding new ones (i.e., attention layers, deconvolutional layers, autoencoders). On the other hand, architecture simplification models apply rule extraction approach to generate human interpretable rules.

Decision models, instead, deep dive in the decision-making process. Also, decision models can be divided into two subcategories: feature relevance models and visual models. These two categories

would be the focus of the practical application in chapter 3. Among decision models techniques, the most relevant are Gradient-weighted Class Activation Mapping (Grad-CAM) and Saliency Maps.

Grad-CAM is a technique introduced to enhance the interpretability of CNNs by providing insights into the regions of an image that influence the network's predictions. Grad-CAM was first introduced in the paper titled "*Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization*" (Ramprasaath R. Selvaraju, 2017). Prior to Grad-CAM, techniques like Class Activation Mapping (CAM) were used to visualize CNNs, but they were limited to specific network architectures, especially those with global average pooling layers. Grad-CAM overcame this limitation by working with any differentiable CNN architecture, making it a versatile tool for interpreting deep learning models. In fact, unlike CAM, Grad-CAM utilizes the gradient information flowing into the last convolutional layer of the CNN to capture the importance of each feature map for a specific class. By computing the gradients of the target class with respect to the feature maps, Grad-CAM generates a localization map, which is then weighted by the gradient values to produce the final heatmap. More specifically, its functioning can be summarized by the following steps:

- Forward Pass: The input image is fed forward through the CNN until it reaches the last convolutional layer.
- Gradient Calculation: Grad-CAM computes the gradient of the score corresponding to the target class with respect to the feature maps of the last convolutional layer.
- Global Average Pooling: The gradients are globally averaged to obtain the importance weights for each feature map.
- Weighted Combination: The importance weights are used to compute a weighted combination of the feature maps, resulting in a rough localization map.
- ReLU and Upsampling: The rough localization map is passed through a ReLU function to retain only the positive contributions, followed by upsampling to match the size of the input image.
- Heatmap Generation: The upsampled map is combined with the original input image to generate the final heatmap, where brighter regions indicate higher importance for the target class.

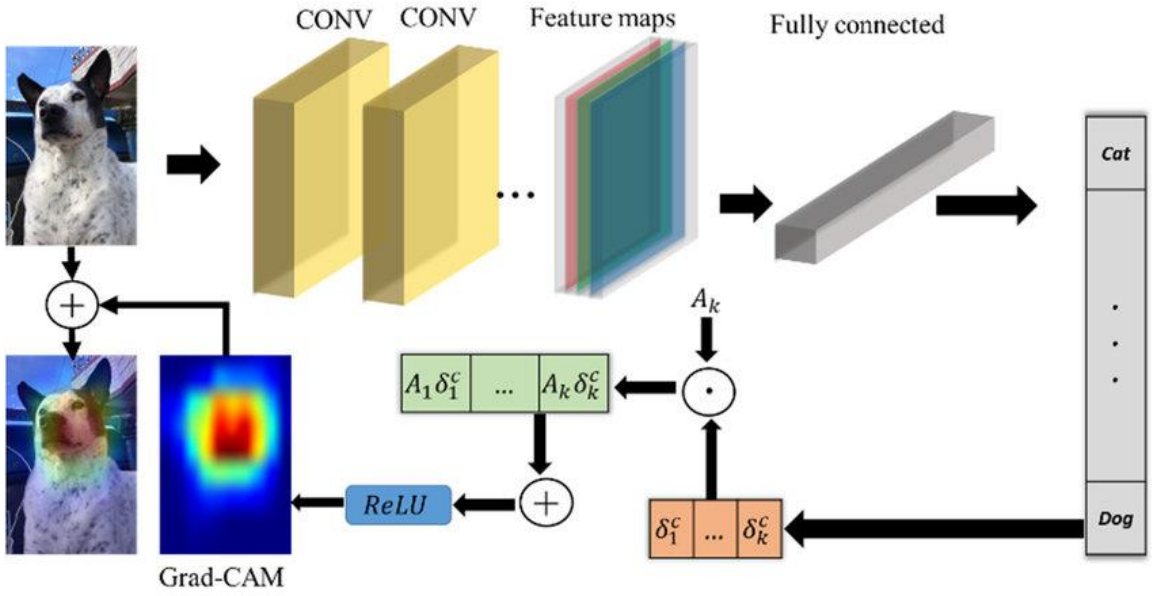

Figure 7: Grad-CAM functioning example,

source https://www.researchgate.net/figure/Grad-CAM-architecture_fig3_352795278

Since its introduction, Grad-CAM has been widely adopted and extended in various domains, including medical imaging, autonomous driving, and natural language processing. Moreover, researchers have explored the application of Grad-CAM in novel contexts, such as multimodal learning, where it can provide insights into the fusion of information from different modalities.

Recent implementations have focused on improving the interpretability and robustness of Grad-CAM. The first to be introduced was Smooth Grad-CAM (Daniel Smilkov, 2017). Smooth Grad-CAM aims to reduce noise in the generated heatmaps by averaging multiple perturbed images during the gradient calculation process. Instead of computing gradients from a single image, Smooth Grad-CAM averages gradients obtained from multiple noisy versions of the input image. By averaging gradients over multiple perturbed images, Smooth Grad-CAM provides more stable and visually coherent heatmaps compared to the original Grad-CAM. This is particularly useful in practical application where the stability and reliability of the heatmap visualization are crucial, such as, for instance, in medical imaging. In 2018, instead, Grad-CAM++ was proposed in the paper "*Grad-CAM++: Generalized Gradient-based Visual Explanations for Deep Convolutional Networks*" (Chattopadhay, 2018). Grad-CAM++ extends the original Grad-CAM by incorporating both positive and negative gradients to improve localization accuracy. By incorporating all gradients in absolute value, Grad-CAM++ captures more nuanced information about the contribution of each feature map to the target class prediction, resulting in effective heatmaps' interpretation.

Other two remarkable Grad-CAM variants, despite their recentness, are Grad-CAM with Attention Mechanisms, and Multimodal Grad-CAM. The first one integrates attention mechanisms with Grad-CAM, improving its interpretability by dynamically highlighting relevant regions. It was proposed in 2023 by Li et al., in the seminal paper "*Enhancing Interpretability of Autonomous Driving Systems with Grad-CAM and Attention Mechanisms*". Attention mechanisms, commonly used in natural language processing (NLP) tasks, allow models to dynamically focus on different parts of the input sequence while processing it. In the context of CNNs, attention mechanisms can be applied to visualize the regions of an image that the network attends to during inference. Grad-CAM with attention mechanisms integrates attention scores obtained from intermediate layers of the CNN with the gradient-based localization provided by Grad-CAM. Instead of solely relying on gradients from the last convolutional layer, attention scores from intermediate layers capture hierarchical features and semantic information, allowing for a more comprehensive understanding of the image. The attention scores serve as weights that modulate the importance of different feature maps in generating the final heatmap. This dynamic modulation enables Grad-CAM to highlight relevant regions more effectively, especially in complex scenes or ambiguous cases.

On the other hand, Multimodal Grad-CAM derives from the paper "*Exploring Multimodal Fusion with Grad-CAM in Deep Learning Models*" (Chen, L., et al. 2023). In multimodal learning, information from different modalities is combined to make predictions or solve tasks. For example, in image captioning, a model may use both an image and its corresponding text description to generate a caption. Multimodal Grad-CAM extends Grad-CAM to interpret multimodal learning models. It generates heatmaps for each modality separately and then combines them to provide a holistic understanding of the model's decision-making process. For example, in a multimodal image classification task, Multimodal Grad-CAM would generate separate heatmaps for the image modality and the text modality. These heatmaps would highlight the important regions in the image and the key words in the text that influence the model's prediction.

For what concerns Saliency Maps, the concept of saliency maps first emerged in the paper titled "*Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps*" (K Simonyan, 2013). Since then, saliency maps have become a popular method for visualizing the regions of an image that are most relevant to a model's prediction. Saliency maps are generated by computing the gradient of the output score with respect to the input image pixels, indicating which pixels have the greatest influence on the model's prediction. For this reason, resulting maps provide a general overview of the input image's salient regions without specifying the particular features or

regions that drive the model's prediction. While they offer insights into which areas of the image the model pays attention to, they may lack the specificity and localization provided by Grad-CAM.

Most notable Saliency Maps modification is Integrated Gradient Saliency Maps. Integrated Gradient was introduced in the paper titled "*Axiomatic Attribution for Deep Networks*" (Mukund Sundararajan, 2017). This paper presents the Integrated Gradients method as a technique for providing better understanding of deep neural networks, offering more robust and reliable explanations of model predictions. This integration helps capture the model's behavior across different input space regions and improves interpretability. Integrated Gradient is applied to the computation of saliency maps. Instead of computing gradients with respect to individual input pixels, Integrated Gradient computes the average gradient along the straight path from a baseline input (e.g., an image with all pixels set to zero) to the actual input image. These gradients are then averaged to obtain the final saliency map, which highlights the regions of the input image that contribute most to the model's prediction. They provide insights into the features and regions of input data that influence model predictions, aiding in model debugging, validation, and trustworthiness assessment.

In conclusion, CNNs represent powerful tools for various applications, yet their intrinsic "black box" nature often obscures their decision-making processes, leading to challenges in interpretability. Consequently, a plethora XAI techniques have been developed and investigated to shed light on CNN implementations. Model agnostic techniques, exemplified by LIME, offer broad applicability but lack the depth of insights provided by model-specific approaches. Conversely, the literature on model-specific techniques is well-structured, notably categorized into decision models and architecture models. Decision models delve into the decision-making process, focusing on feature relevance and visual interpretations, while architecture models analyze the network structure. Among these, Grad-CAM stands out as a pivotal technique, revolutionizing CNN interpretation by generating visual explanations based on gradient information. Its versatility and robustness have been further enhanced through subsequent variants like Grad-CAM++, Grad-CAM with Attention Mechanisms, and Multimodal Grad-CAM. Saliency maps, introduced earlier, offer a broader overview of salient image regions but lack the localization precision of Grad-CAM. Nevertheless, recent modifications like Integrated Gradient Saliency Maps have enriched their interpretability and robustness. The vastity and recentness of the abovementioned literature significantly highlight the sparkling vivacity and room for innovation which characterize the field of XAI applied to CNNs.

3. Practical Application: Areal Images Scene Classification

3.1 Introduction

The classification task regarding scenes in areal or satellite images represents an interesting and significant research area within the field of computer vision. This phenomenon is fostered by an increasing availability of high-quality areal images and the growing need for precise and accurate applications addressing the earth observations issue.

This paragraph deals with the definition, background and current state-of-the-art methodologies, together with practical applications, laying the basis for the exploration of model implementation and analysis.

Scene classification implies the analysis of satellite and areal imagery, in order to divide them in specific setting categories. This definition, obviously, lays down on principles of image processing, machine learning and deep learning to interpret and understand input images accurately. This process is pivotal in commuting raw input, in the form of satellite images, into insights for environmental monitoring, urban planning, disaster management, and agricultural assessment, among other fields.

The evolution of image classification techniques consists in transitioning from traditional techniques, like for example likelihood classification and decision trees, to more complex and sophisticated methods such as Convolutional Neural Networks (CNNs), which leverage deep learning for enhanced accuracy and efficiency in handling complex image datasets (Kaiming He, 2016).

A numerous and heterogeneous set of machine learning methods have been employed in the field of satellite image classification to leverage complex datasets in order to extract meaningful information. These methods span from traditional methods to advanced deep learning techniques, each of them offering different and unique strengths in handling and solving this type of task. Support Vector Machines (SVMs), among simpler approaches, have been employed thanks to their effectiveness in high-dimensional data classification tasks, leveraging a margin-based technique to distinguish between land cover types (Melgani & Bruzzone, 2004).

Decision Trees and Random Forests strengths, instead, consist in their interpretability power and ability to handle non-linear data, using ensemble learning to reach great results in term of classification accuracy (Ghimire, 2012).

K-Nearest Neighbors (KNN) by analyzing the spatial proximity of data points, represents another solid approach, which has its strengths in its simplicity and effectiveness. Despite that, the introduction of CNNs has revolutionized the field, providing unmatched accuracy in

scene recognition tasks. CNNs are capable to automatically detect and learn hierarchical patterns, and this makes them perfectly suited for processing and classifying spatial and spectral complexity of satellite imagery (Kaiming He, 2016). These neural nets have demonstrated relevant success in classifying images based on learned features, rather than relying on hand-crafted ones. Other techniques, addressing a slightly different problem, are Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM). These are employed for analyzing temporal changes in satellite data and offer information about more dynamic phenomena like vegetation growth or urban development over time.

Coming back to CNNs, a great variety of techniques specifically tailored for satellite image classification were introduced. Among them, several have risen to prominence, thanks to their outstanding performances, paired with an exceptional adaptability to different tasks and datasets. U-Net (Ronneberger, 2015) is for sure one of them. It was originally designed for biomedical image segmentation, but, as it commonly happens in those cases, was widely adopted for its efficiency in handling satellite imagery with its symmetric expanding and contracting paths that enable precise localization.

Another relevant CNN approach is the Residual Network (ResNet), which introduced for the first time the concept of residual learning to address the vanishing gradient problem, allowing neural networks to employ more layers and significantly improve classification accuracy (Kaiming He, 2016). This feature makes ResNet particularly effective for processing the high-dimensional data typical of satellite images.

Another technique called SegNet (Segmentation Network), characterized by a different approach consisting in a pixel-wise classification, also gained great relevance in the literature, thanks to the ability to perform semantic segmentation of satellite images, critical for detailed land cover and land use mapping (Badrinarayanan, Kendall, & Cipolla, 2017). In addition to that, the Dense Convolutional Network (DenseNet) architecture is celebrated for its dense connectivity pattern. It enhances feature propagation and, at the same time, reduces the number of parameters, making DenseNet for satellite image analysis efficient and powerful at the same time (Geoff Pleiss, 2017).

All the aforementioned machine learning methods bring a unique perspective to approach the challenge of satellite image classification, providing researchers and practitioners different ways to tackle the analysis and understanding of earth observation data. Choosing the right method depends on the specific application, dataset characteristics and desired accuracy, highlighting the importance of having a tailored approach.

These CNNs implementations have been propaedeutic for satellite image classification progresses, offering a robust and consistent framework for extracting meaningful insights from areal imagery. Their diffusion stems not only from their performances but mainly from their flexibility and adaptability to a wide range of applications. The practical applications of areal images classification extend across numerous fields, demonstrating its versatility and potentiality.

In the field of environmental monitoring, satellite image classification serves as an essential tool for a lot of stakeholders, that can vary from scientists to policymakers. Its relevance is due to its ability to enhance the detailed observation of global ecosystems over time, providing significant data for tracking relevant events. An example can be deforestation rates, a phenomenon with far-reaching implications for biodiversity, climate change, and human livelihoods (Hansen, 2013). Furthermore, this technology is pivotal in monitoring the status of natural ecosystems, including wetlands, forests, and coral reefs, offering insights into the effects of environmental stressors and human activities. The ability to observe changes in land use and land cover over wide areas and extended periods is essential for informed decision-making and effective conservation strategies.

Moreover, urban planning and development benefit significantly from the insights gained through areal image classification. By enabling detailed analysis of urban expansion, these machine learning approaches assist in the sustainable management of urban growth, helping to balance development needs with environmental considerations (Xu & Fortes, 2010). In fact, the classification of satellite imagery aids in natural resources' management within urban areas, facilitating green space planning, water resource management, and urban heat island mitigation efforts.

In agriculture, the classification of areal images has revolutionized the way in which crop management is approached. By identifying crop types and estimating yields, farmers and agricultural researchers can optimize resource allocation, enhance productivity, and minimize environmental impact (Mulla, 2013). Precision farming practices, supported by satellite image classification, enable the targeted application of water, fertilizers, and pesticides, reducing waste and increasing efficiency. This technology also supports soil health monitoring and the management of agricultural diseases and pests, contributing to more sustainable and resilient farming systems.

Following natural disasters, the rapid classification of satellite imagery becomes a critical component of the response and recovery efforts. It allows for the quick assessment of damage to infrastructure and natural landscapes, facilitating effective resource allocation and prioritization of recovery activities (Divyani Kohli, 2012). The insights gained from areal image classification can guide

emergency response teams, aid distribution efforts, and long-term rehabilitation plans, ultimately reducing the impact of disasters on affected communities.

The ongoing advancements in areal images classification methods, particularly through the adoption of machine learning and deep learning, promise to further enhance the precision, efficiency, and scope of these applications. For example, the integration of multi-temporal and multi-spectral data enables detailed monitoring of dynamic processes and phenomena, while the development of more sophisticated algorithms improves the accuracy and reliability of classification outcomes. As these technologies continue to evolve, they hold the potential to address some of the most pressing challenges faced by society, from climate change and environmental degradation to urban development and food security, demonstrating the transformative power of areal image classification in shaping a more sustainable and resilient future.

3.2 Dataset

3.2.1 Dataset Overview and Previous Use Cases

This paragraph will provide complete overview of all relevant datasets in the scene classification field, together with a brief description of the mentioned datasets, including modification and motivations. Several such datasets have become benchmarks in the field of scene classification, especially in evaluating performances of machine learning and deep learning models. These datasets vary across multiple features, such as size and complexity, or for types of scenes they contain, offering users diverse resources for training and testing algorithms. The most common datasets used for scene classification are the following, mentioned in chronological order:

1. UC Merced Land Use Dataset: published in 2010, this dataset contains 2100 aerial scene images categorized into 21 land use classes, with 100 images each. The images are taken from the USGS[4] National Map Urban Area Imagery collection for various urban areas around the US. The dataset is widely used for its diversity in urban, agricultural, and natural landscapes (Newsam, 2010).
2. NWPU-RESISC45 Dataset: published in 2017, the RESISC45[5] dataset, provided by the Northwestern Polytechnical University (NWPU), consists of 31500 images, covering 45 scene classes with 700 images each. RESISC stand for Remote Sensing Image Scene Classification. This dataset was proposed in a paper called "*Remote Sensing Image Scene Classification: Benchmark and State of the Art*" (Cheng, Han, & Lu, 2017), where the authors define its best

[4] United States Geological Survey
[5] Remote Sensing Image Scene Classification

features as “1) large-scale on the scene classes and the total image number; 2) holds big variations in translation, spatial resolution, viewpoint, object pose, illumination, background, and occlusion; and 3) has high within-class diversity and between-class similarity”.

3. AID (Aerial Image Dataset): published in 2017, the AID dataset is a large-scale dataset for aerial scene classification containing more than 10000 images divided into 30 classes. It was proposed in a paper called “*AID: A Benchmark Dataset for Performance Evaluation of Aerial Scene Classification*” (Xia, et al., 2017), with the aim to solve previous limitations in the field. It constitutes a big improvement with respect to UC Merced Land Use Dataset, but it is still smaller than NWPU-RESISC45 Dataset.
4. EuroSAT: published in 2017, it was introduced in the paper “*EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification*” (Helber, Bischke, Dengel, & Borth, 2017). It covers 13 spectral bands and consists of 10 classes with 27000 labeled and geo-referenced images, based on Sentinel-2: “The Sentinel-2 satellite images are openly and freely accessible provided in the Earth observation program Copernicus” (Helber et al. 2017). The novel dataset was designed for land use and land cover classification and it is particularly valuable for its spectral diversity.
5. Google Earth Engine (GEE) Data Catalogue: Earth Engine's public data archive includes more than forty years of historical imagery and scientific datasets, updated and expanded daily. While not a single dataset, Google Earth Engine provides access to a vast collection of satellite imagery and geospatial datasets that can be leveraged for custom scene classification tasks. GEE's extensive library includes Landsat, Sentinel, and MODIS datasets, among others, offering unprecedented access to global satellite data.
6. DeepGlobe Land Cover Classification Challenge Dataset: part of the DeepGlobe Satellite Challenge, this dataset includes satellite images for land cover classification across multiple categories such as urban, agriculture, rangeland, water, etc. It's designed to advance research in geospatial analysis and it “aims at bringing together a diverse set of researchers to advance the state-of-the-art in satellite image analysis.” ((http://deepglobe.org), s.d.)

All the aforementioned datasets have played a pivotal role in the development and evaluation of scene classification algorithms, providing different challenges that reflect real-world complexity and diversity. This wide set of available databases results in the possibility to leverage these resources to benchmark accuracy, explore new and diverse methodologies, contributing in advancing the state-of-the-art in satellite image analysis and classification.

3.2.2 Description of the Model's Dataset

The dataset chosen for the purpose of this project is NWPU-RESISC45 dataset. This choice comes from several reasons. The first one relies on the fact that it is the widest used dataset, and this provides a great variety of benchmarks to look at while analyzing the performances of different deep learning models. The second reason is strictly related to the dataset volume. In fact, NWPU-RESISC45 provides to the users a great variety of classes, and each of them has a sufficient size to train properly the model. Most recent datasets, such as DeepGlobe or Google Earth Engine were excluded from this dissertation because they offer a shorter background literature, and have more complex nature with respect to the scope of a simple image classification tasks.

The scope of this machine learning application is to interpret as much as possible both the architecture and the output of the neural networks involved in the process. In order to perform a clear and comprehensive evaluation of the two aforementioned aspects, some limitations must be imposed to the dataset. While the architecture is just tangentially affected by this choice, the output interpretability is highly dependent on the data, especially for what concerns the number of classes. As highlighted before, NWPU-RESISC45 covers 45 scene classes. A great number of classes could create problems while visualizing and interpreting the results, generating messy and hard to follow plots and tables. The best approach to solve this issue consists in reducing the number of classes that would be considered in the analysis. Therefore, the final dataset is composed by the ten most relevant categories of images, which are the following: i) wetland, ii) terrace, iii) snowberg, iv) see ice, v) river, vi) mountain, vii) meadow, viii) forest, ix) desert, and x) beach. The criteria adopted focus primary on the categories more relevant from an environmental point of view. Choosing different classes would not impact the validity or the results of this dissertation, but the coherence adopted while filtering the classes has a vital importance in giving to this project all the characteristics of a real-word scenario application.

3.3 Models Selection

This section provides a deep dive into the literature review and architecture of the two models selected to perform scene image classification on a subset of the NWPU-RESISC45 dataset. The chosen models are ResNet and DenseNet. The first aspect needing further explanation is the decision to investigate two different models. Implementing both ResNet and DenseNet allows for comparison between diverse architectures and outputs, highlighting the power of Explainable AI techniques. Another aspect requiring clarification is the actual choice of which CNNs models are best suited for this thesis's purpose. Among the criteria applied to make this decision there are: i) the presence of literature providing benchmarks for the results, ii) the quality of results in terms of accuracy, and iii)

structural diversity to enable a more meaningful analysis of differences and commonalities. Considering all these aspects, ResNet and DenseNet constitute an excellent starting point. This is because, as will be further explored in the following sections, both have been widely discussed in a considerable number of papers in recent years. These papers themselves have pointed out outstanding performances on almost all the aforementioned datasets containing satellite images. In conclusion, regarding structure and architecture, all differences will become clear by the end of this chapter.

3.3.1 ResNet and DenseNet landscape

ResNet, short for Residual Network, is a deep CNN architecture introduced in their paper called "*Deep Residual Learning for Image Recognition*", published in 2016 (Kaiming He, 2016). This paper introduces the ResNet architecture, proposing residual connections to address the problem of vanishing gradients, referred to as the “degradation problem” in the paper, that characterizes extremely deep neural networks. The “degradation problem” occurs when the depth of neural networks increases and the accuracy becomes saturated, and therefore degrades rapidly. As Kaiming He et al. (2016) explain in the paper, "such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error." The authors address the degradation problem by introducing a deep residual learning framework, as explained in the architecture section. The introduction of this novel approach to CNNs has had an incredible impact on research in this field, sparking a significant debate involving numerous new applications and modifications to the proposed Residual Network architecture. In fact, ResNet models showed superior performance on various benchmark datasets like ImageNet, CIFAR-10, and COCO.

For what concerns its applications, ResNet algorithms have been employed in the most disparate computer visions fields, including, but not limited to, object detection, semantic segmentation, and image generation. For example, in object detection, the paper "*R-FCN: Object Detection via Region-based Fully Convolutional Networks*" by Jifeng Dai (2016) states clearly that "the incarnation of R-FCN in this paper is based on ResNet-101." In the field of semantic segmentation, "*Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes*" (Tobias Pohlen, 2017) proposes "a novel ResNet-like architecture that exhibits strong localization and recognition performance." For image generation applications, an exemplifying paper is "*Pose-Normalized Image Generation for Person Re-identification*" (Xuelin Qian, 2018), where images are fed to a ResNet architecture after normalization.

Besides the wide range of applications, numerous new models derived from ResNet have been developed. In the already cited paper "*Deep Residual Learning for Image Recognition*" (Kaiming He, 2016), the authors proposed several variants of ResNet, including ResNet18, ResNet34, ResNet50,

ResNet101, ResNet152, and even deeper versions. These variations differ in their depth, with the number indicating the number of layers present in the neural network. Other authors, in other papers, attempted to leverage the residual learning innovation, proposing substantial architectural modifications. Among the most adopted are Wide Residual Networks (WRN), aggregated residual transformations for deep neural networks (ResNeXt), and densely connected convolutional networks (DenseNet). Wide Residual Networks (WRN) were introduced in in the homonymous paper "*Wide residual networks*" (Zagoruyko & Komodakis, 2016). The authors introduced it because, according to them, "each fraction of a percent of improved accuracy costs nearly doubling the number of layers, and so training very deep residual networks has a problem of diminishing feature reuse, which makes these networks very slow to train" (Zagoruyko & Komodakis, 2016). This consideration results in novel neural networks, whose architecture relies on ResNet but with decreased depth and increased width. ResNeXt, instead, was proposed in the paper "*Aggregated Residual Transformations for Deep Neural Networks*" (Saining Xie, 2017). The authors introduced "a simple, highly modularized network architecture for image classification," which is "constructed by repeating a building block that aggregates a set of transformations with the same topology" (Saining Xie, 2017). The size of the set of transformations is called "cardinality," which, together with depth and width, constitutes a new set of dimensions of ResNeXt. In conclusion, the DenseNet model will be analyzed more deeply since it will be implemented together with ResNet. Densely connected convolutional networks were introduced for the first time in the homonymous paper published by 2017 (Gao Huang, 2017). The underlying assumption, empirically demonstrated by other recent works, of the whole paper is that convolutional networks can be "substantially deeper, more accurate, and efficient to train if they contain shorter connections between layers close to the input and those close to the output" (Gao Huang, 2017). According to the authors and their findings, "DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters" (Gao Huang, 2017). Also, the DenseNets innovation spillover effect led to its implementation in different fields. A great example is the paper "*The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation*" (Simon Jegou, 2017). By expanding and adapting DenseNets, they obtained outstanding results in most common semantic segmentation dataset such as CamVid and Gatech. In fact, "due to smart construction of the model" their model "has much less parameters than currently published best entries for these datasets", while in the meantime achieving "state-of-the-art results" on the aforementioned urban scene segmentations dataset (Simon Jegou, 2017). As ResNet, during past few years, a lot of modifications were proposed to further enhancing the efficiency in terms of both accuracy and computational expense. The most important is DenseNet-BC. The DenseNet-BC

(DenseNet with Bottleneck and Compression) variant was proposed by the authors of the original DenseNet paper. It introduces bottleneck layers to reduce the number of parameters and improve computational efficiency. Additionally, it incorporates compression by using a reduction factor to decrease the number of feature maps in each transition layer (Gao Huang, 2017).

In conclusion, both ResNet and DenseNet models have not only revolutionized the field of computer vision but also fostered innovations and modifications across various domains. Theirs impact extends beyond image classification, reaching into object detection, semantic segmentation, and image generation, among others. These advancements demonstrate the enduring influence and adaptability of ResNet and DenseNet in addressing contemporary challenges in deep learning and computer vision research. In the next sections, the actual architecture of ResNet and DenseNet will be analyzed and explained.

3.3.2 ResNet Architecture

The core idea of ResNet architecture is the introduction of the residual (or building) block, which adds a shortcut connection that skips one or more layers. In a traditional neural network, each layer learns representations of the input data. In contrast, in a ResNet, each residual block learns the residual function relative to the layer inputs. Mathematically, if $H(x)$ is an underlying mapping to be learned by a few stacked layers, and x is the input, these layers in a ResNet try to learn the residual function:

$$F(x) = H(x) - x$$

The original function thus becomes:

$$H(x) = F(x) + x$$

This is because the authors “hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers” (Kaiming He, 2016). This is implemented by using shortcut connections, also known as “skip connections”, that perform identity mapping, and their outputs are added to the outputs of the stacked layers. The above descripted “block” is represented by the figure below, where the arrow represents the “skip connection”.

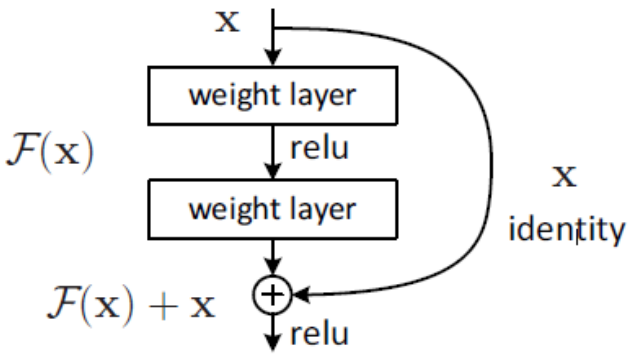

Figure 8: ResNet Residual Block

Formally, the paper's authors consider a building block defined as:

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

Here x and y are input and output vectors of the layers considered. The function

$$F(x, \{W_i\})$$

represents the residual mapping to be learned (Kaiming He, 2016). Always referring to the figure, the operation is performed by a shortcut connection and element-wise addition. The most used shortcut connection simply performs an identity mapping, and its output is added to the output of the stacked layers. If the dimensions are not the same, projections can be used to match dimensions. This is because in the paper was shown "by experiments that the identity mapping is sufficient for addressing the degradation problem and is economical, and thus a square matrix is only used when matching dimensions." The last step to build the actual Convolutional Neural Network is determining the number of layers and the magnitude of the transformation performed by each of them.

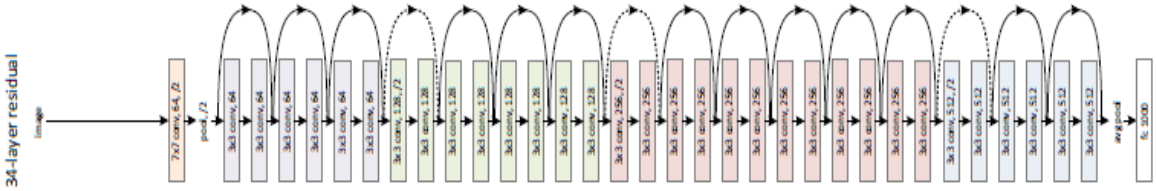

Figure 9: ResNet34 architecture

The Figure 9 above shows how a ResNet34, i.e., with 34 convolutional layers, looks like. The basic architecture of the neural is "inspired by the philosophy of VGG[6] nets" (Kaiming He, 2016). This means that the convolutional layers predominantly utilize 3x3 filters and is compliant to two fundamental design principles: firstly, layers that produce feature maps of the same size have an equal number of filters; and secondly, when the size of the feature map is halved, the number of filters is doubled to maintain the same time complexity per layer. Down-sampling, indeed, is achieved directly through convolutional layers with a stride of 2. The architecture ends with a global average pooling layer, followed by a fully-connected layer with 1000 outputs and a soft-max function. Overall, the network, as aforementioned, is composed by 34 layers with weights. Building on the described basic network structure, shortcut connections were introduced (as shown in Figure 2), transforming the network into its residual variant. These identity shortcuts are applicable without modifications when the dimensions of the input and output. In cases where dimensions expand, indicated by dotted line shortcuts, the authors evaluated two approaches: "(A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (B) The projection shortcut in Equation (2) is used to match dimensions."

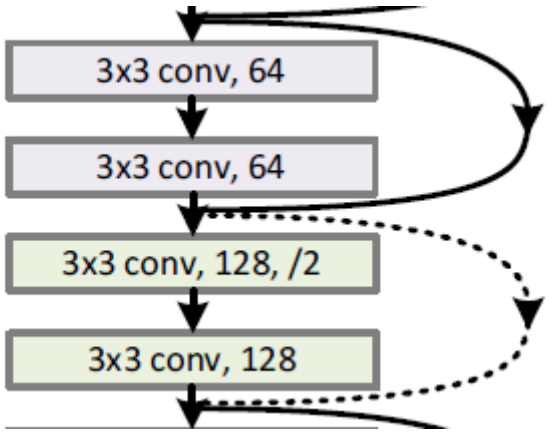

Figure 10: Dimensions change in building blocks

Figure 10 above shows two different "building blocks." The first one, i.e., the one in purple, keep the output as the same form of the input. The second one, i.e., the one in light green, transforms the input size, specifically from 64 to 128. Since that, the first "skip connection" (continuous line) is the identity one, while the second one could be both the identity mapping (Option A) or a projection matrix (Option B).

[6] Visual Geometry Group

Until now, the number of layers considered was 34, but training significantly deeper neural networks, like ResNet50 or ResNet101, implies an exponential growth of model's complexity and computational cost. Bottleneck layers serve as a strategic component in deep neural networks in order to solve this issue. The rationale behind the use of bottleneck layers is to efficiently manage the network's depth, ensuring that it can learn more complex features without a proportional increase in computational cost and training difficulty. The structure of a bottleneck layer is meticulously designed to reduce the flow of information through the network while preserving the integrity of the input's feature representation. This design is realized through a sequence of three distinct layers:

1. Dimensionality Reduction (1x1 Convolution Layer): The first layer in a bottleneck structure employs 1x1 convolutions. Despite its simplicity, this layer performs a critical function by reducing the dimensionality of the input feature maps. By doing so, it significantly decreases the number of input channels to the subsequent layer, thus reducing the computational load. This layer essentially compresses the input information, making it more manageable for the network to process, without losing the essence of the feature representation.
2. Feature Processing (3x3 Convolution Layer): Following the dimensionality reduction, a 3x3 convolution layer takes over to process the now-compressed features. The 3x3 convolutions are applied to the reduced feature maps, enabling the network to extract spatial features efficiently. This layer benefits from the reduced dimensionality, allowing it to focus on understanding the spatial relationships within the data with fewer parameters and lower computational cost compared to processing the original, high-dimensional feature maps.
3. Dimensionality Restoration (1x1 Convolution Layer): The final layer in the bottleneck design is another 1x1 convolution layer, which serves the opposite purpose of the first. Instead of reducing, it restores the dimensionality of the feature maps to their original size. This step is crucial for integrating the bottleneck block into the larger network architecture, ensuring that the output can seamlessly connect with subsequent layers or blocks.

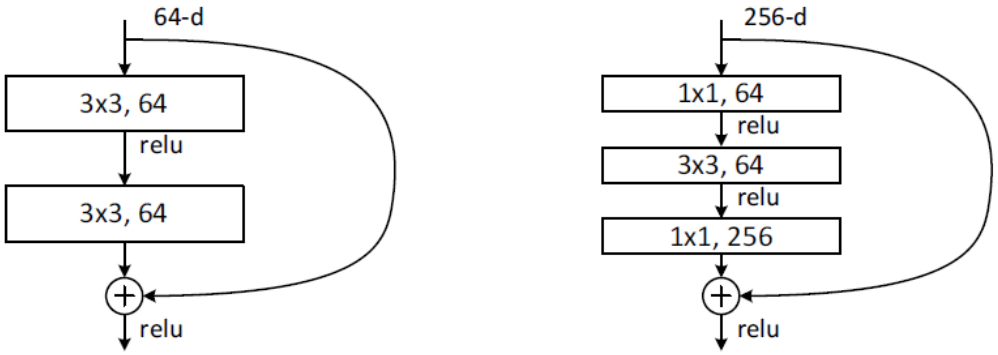

Figure 11: Bottleneck layers

The brilliance of the bottleneck design lies in its ability to reduce, process, and then restore the information flow within the network in a highly efficient manner. This approach enables very deep networks to be trained more effectively and efficiently, as it mitigates the rapid increase in computational requirements that would otherwise occur with increased depth. Bottleneck layers, therefore, represent a sophisticated balance between performance, computational efficiency, and the ability to train deep models capable of learning highly complex features.

Finally, the following Figure 12 shows the architecture of ResNet101. Of the 101 layers, 99 come from the 33 "bottleneck residual blocks." There are, in fact, 3 building block of type conv2, 4 of type conv3, 23 of type conv4 and 3 of type conv5. There are two types of pooling performed: max pooling at the beginning and average pooling at the end.

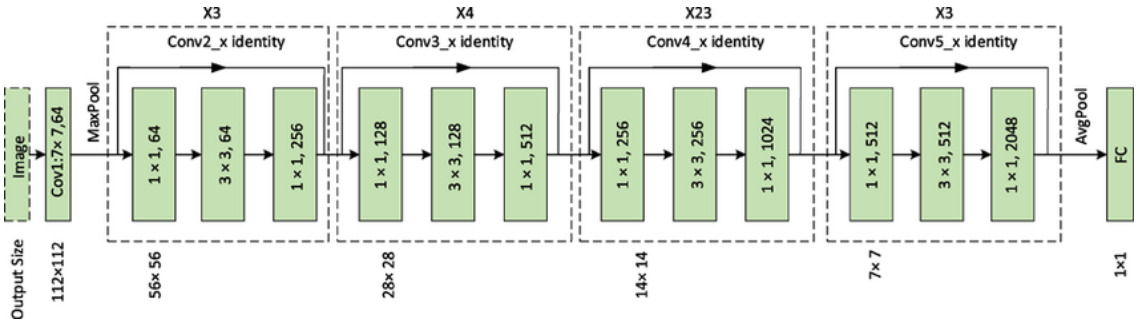

Figure 12: ResNet101 architectures

3.3.3 DenseNet Architecture

DenseNet core innovation consists in its dense connectivity pattern, a significant departure from traditional convolutional network architectures. This design emphasizes improving the flow of information and gradients throughout the network, which facilitates training deeper networks and

improves efficiency and performance. In DenseNet, “each layer is connected to every other layer in a feed-forward fashion” (Gao Huang, 2017). For a traditional convolutional network with L layers, there would typically be L connections, one between each layer and its subsequent layer. In contrast, a DenseNet with L layers has

$$n = L(L + 1)$$

connections. This means for each layer, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers. This approach relies on ResNet intuition, but changes the connectivity flow, adding a considerably greater number of forward connections. What in ResNet was called “building block”, in DenseNet become a “dense block”. A “dense block”, like shown in Figure 14, is composed by a set of convolutional layers, generally in the form of BN-ReLU-Conv [7] , where each of them is connected with all the layers after it. In this setup, each “dense block”, comprises a batch normalization layer, followed by a 1x1 convolutional layer, and concludes with a 2x2 average pooling layer to facilitate the reduction in feature-map size.

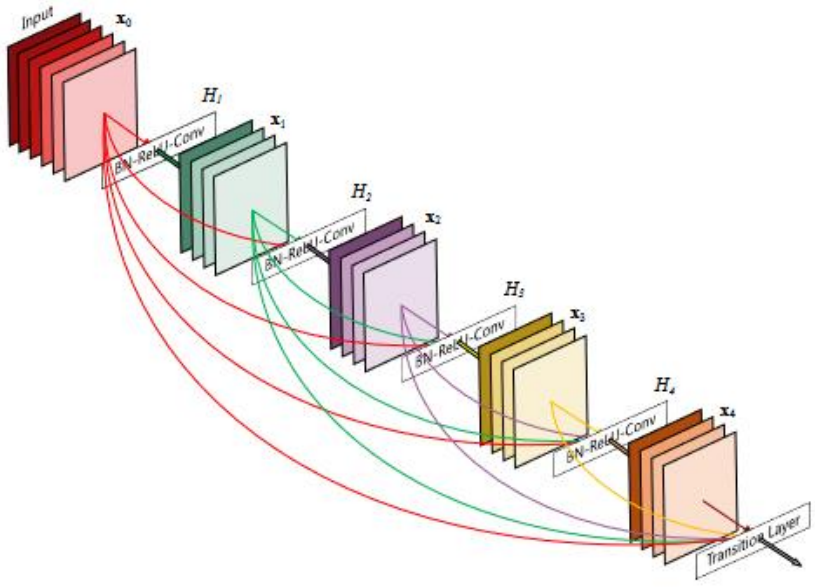

Figure 13: General DenseNet architecture

In DenseNet network design, directly concatenating feature becomes impractical when the dimensions of the feature maps vary. To effectively incorporate down-sampling within the model,

[7] Batch Normalization, ReLu activation and Convolution

network architecture is organized into several densely connected blocks. The layers situated between these dense blocks are known as “transition layers”, responsible for performing both convolution and pooling operations. The concatenation of two “dense blocks”, happens as showed in Figure 14.

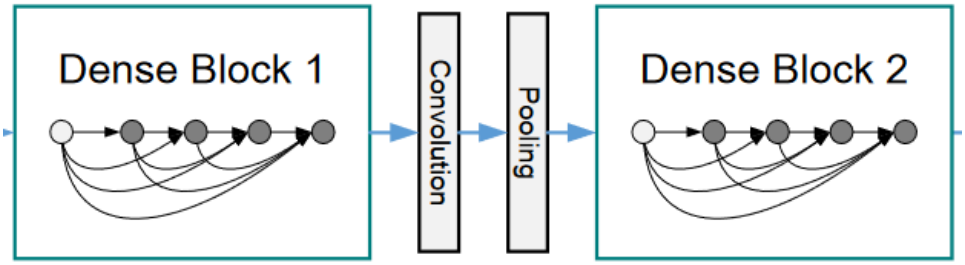

Figure 14: DenseBlock architecture and concatenation

Within this kind of DenseNet architecture, each layer, denoted as ℓ, produces k feature maps. This means that the ℓ^{th} layer receives

$$k_0 + k(\ell - 1)$$

input feature maps, where k_0 represents the number of channels in the first layer. A distinctive feature of DenseNet, compared to other network architectures, is its ability to have very “narrow layers”. For instance, in DenseNet are typically present layers with k of just 12. The term "growth rate", in this framework, refers to the parameter k, which signifies the amount of new information each layer contributes to the network's overall knowledge. This is due to the fact that, unlike traditional architectures, in DenseNet, every layer can access all previous feature maps within its block. As new layers add their own k feature maps, they are added to this global state. The advantage here is that the global state is accessible without the need to duplicate information across layers. Similarly to what happens in ResNet, also in DenseNet the computational expenditure must be taken into considerations. To address this issue, bottlenecks layers, such as the ones present in ResNet, are introduced. In fact, “though each layer only produces k output feature-maps, it typically has many more inputs, […] 1×1 convolution can be introduced as bottleneck layer before each 3×3 convolution to reduce the number of input feature-maps, and thus to improve computational efficiency” (Gao Huang, 2017).

Finally, the DensNet used for the scope of this application is composed as follows:

- 3 dense blocks, composed by a set of three operations each: Batch Normalization, ReLU Activation and Convolution (3x3).
- 4 transition blocks, composed by a set of four operations each: Batch Normalization, ReLU Activation, Convolution (1x1) and Pooling.

- A growth rate of k =12.

3.4 XAI Techniques Interpretation

The XAI techniques that will be applied are: i) visualizing Filters and Feature Maps, ii) Grad-Cam, iii) Saliency Maps and iv) LIME explainer. Their functioning was already widely discussed and explained in previous sections; however, the utility of these methods extends beyond technical details. Indeed, it lies in the ability to correctly interpret the visualizations they generate. Misinterpretation can undermine the very purpose of XAI. This section investigates proper understanding of visual outputs produced by aforementioned XAI techniques applied to both ResNet and DenseNet. By analyzing the nuances of these visualizations, it is possible to comprehend not just how these models process images (e.g., 'see'), but also how they take decisions (e.g., 'think'). In doing so, the main aim is to provide the sufficient knowledge to explain the results' section conclusions.

3.4.1 Filter and Feature Maps

Interpreting the visualizations of filters in a CNN provides insights into how the network processes and perceives input images. These visualizations are the key to understand what features or patterns a network is paying attention to, and, at the same time, what important information may be ignoring. Filter maps shape varies from different layers according to the layers' depth. In fact, early layers filters usually learn to detect basic features such as edges, corners, and colors. If you see filters that highlight edges in various directions or specific colors, it indicates that the network is learning to pick up these fundamental components of the input images. As you move to deeper layers, but not yet to the deepest ones, filters start to specialize in detecting textures and simple patterns, which are the results of combination of edges, corners and colors. For this reason, visualizations from these layers might show more complex patterns, such as grids, stripes, or simple geometric shapes. This suggests that the network is learning to combine basic features into more specific ones, which can be useful for distinguishing between different classes. In the deeper layers of the network, filters try to catch more complex shapes. The visualizations from these layers can be less intuitive to understand because they capture higher-level abstract features. They may look like specific parts of objects or even more abstract patterns that do not directly map to a simple visual feature. These representations are highly specialized to the task the network is trained on and indicate that the network is learning to identify and distinguish between complex objects and scenes.

There are several perspectives from which is possible to interpret filters. The first one is surely looking for progression. In fact, there should be a clear progression from simple to complex features as you move from the early to the deep layers. This progression is a good sign that the network is

building up its understanding of the images in a hierarchical manner. The second one is the identification of specialized features. In the deeper layers, it can be possible to spot what kinds of objects, shapes, or patterns each filter is responding to. This can give you insights into what features are considered important for the decisions the network is making. The third purpose that filters can serve is checking for redundancy. If many filters seem to be detecting very similar features, it might indicate the spreading of repetitive information within the network. While some redundancy can be beneficial for robustness, excessive redundancy might mean the network is not utilizing its capacity efficiently. Through filers it is also possible to search for missing features. If you expect your model to learn certain features but then these features are not represented in the filters, it might indicate a gap in the network's ability to learn. This could be due to insufficient training data, poor network architecture, or inadequate training time. So, checking filters after the modification of one, or all, of the aforementioned issues, could be an efficient way to solve it.

Lastly, analyzing neural networks filters can help to highlight and detect overfitting. If the filters in the deeper layers seem to be too specific, detecting very peculiar or overly detailed patterns that are not generally representative of the class it is supposed to detect, it might be a sign of overfitting. This means the network might perform well on the training data but poorly on unseen data. Interpreting CNN filters can be very difficult, since it is not an “exact science” and it requires both practice and intuition. However, gaining insights into how a network processes images can be incredibly valuable for improving model architecture and understanding the limitations and strengths of your model. Since that, for this specific application, filters would be primary displayed and discussed to provide a better overview of the transformations which are applied to input, helping at the same time to better understand ResNet and DenseNet architectures.

On the other hand, feature maps offer a method that generates more interpretable outputs. These are essentially the outputs from the networks’ convolutional layers, capturing the filters’ responses to the input image. Each feature map emphasizes various aspects of the input image, influenced by the filter distinctive features, such as their position within the network’s depth. Consequently, the interpretation of feature maps is directly linked, and kind of opposite, to understanding the behavior of filters. Therefore, insights regarding the depth of filters layers are also applicable in the context of feature maps interpretation. Bright areas in a feature map indicate regions where the filter’s pattern strongly matches the input image. For example, an edge-detecting filter might produce bright areas along the edges in the image. As you move deeper, feature maps start to represent more complex patterns, textures, or parts of objects. These indicate the types of intermediate features the network is learning to recognize. Feature maps size decreases as you go deeper into the network, focusing on more

abstract and complex features. At this stage, feature maps are less precise about spatial information. In the deepest layers visualization may be more difficult to interpret visually because they capture the essence of the objects in a very abstract way. These maps are focuses more about the conceptual presence of scene elements rather than explicit visual patterns.

Finally, regarding the interpretation, it is crucial to consider all different aspects that affect the visualizations. First, focusing on brightest area highlighted in the feature maps, it is possible to identify areas of high activation. These regions reveal the features that activate specific filters, offering insights into what is the network prioritization mechanism. This approach enhances the understanding of the network's learning process.

Moving forward, is crucial to assess layer by layer differences and similarities. Starting from initial layers, and progressing deeper into the network, it should be evident how each feature map evolves from concrete details to more abstract concepts. This gradual abstraction process is a fundamental characteristic of CNNs, allowing them to go beyond simple pixel values. Moreover, examining the variety across channels within the same layer reveals the network's capacity to capture a wide array of patterns. A well-functioning network will display feature maps that respond to different aspects of the input data, indicating a robust ability to analyze and interpret diverse features. Finally, the impact of architectural choices on the feature maps should be considered too. Different convolution types, the inclusion of pooling layers, and other architectural decisions play a significant role in shaping how the network learns and abstracts information. Understanding these effects can provide deeper insights into the design and functionality of CNNs, enabling more informed choices in network architecture, and potentially leading to enhanced performance and interpretability. In fact, this process is key to diagnosing network performance, understanding model decisions, and guiding improvements in model architecture or training procedures.

3.4.2 Grad-CAM

Grad-CAM, or Gradient-weighted Class Activation Mapping, is a technique for visualizing the regions of input that are important for predictions from CNNs. It helps to understand why a model makes a certain prediction about an image, highlighting the areas that significantly influence the classification. In fact, Grad-CAM provides an insightful visualization output that generates heatmaps to be overlaid on the original image, representing clearly which areas the model considers significant for its decision-making process. These heatmaps range in color from blue to red. Blue stands for regions of low importance to the model's decision; red ones, instead, indicates areas of high importance. Portions of image where the heatmap has a greater concentration offers a window into the model's attention mechanism.

However, a misalignment between the heatmap’s focus and the actual relevant features of an image could reveal deficiencies in the model’s learning ability. If the heatmap puts in evidence irrelevant areas, or fails to delineate the object of interest, it might suggest that the model lacks of understanding relevant features. Such issues can be solved by further model refinement like additional training, data augmentation, or architectural modifications. What is more, Grad-CAM allows for the generation of visualizations across different classes, enabling a comparative analysis of which image parts become significant for each category. This aspect is particularly useful in instances where the model exhibits confusion between classes, revealing the features it might erroneously associate with another class. This last feature would be the foundation of the analysis section involving Grad-CAM. While a powerful tool for interpreting model decisions, Grad-CAM has certain limitations. Localization maps provided by Grad-CAM may not precisely outline the object of interest, often leading to a deceiving approximation of what the model is focusing on. Additionally, the effectiveness of Grad-CAM's visualizations heavily relies on the choice of convolutional layer from which gradients are pooled. The last convolutional layer is typically chosen to achieve the best balance between capturing high-level features and maintaining spatial detail. Finally, it is important to note that high activation in the correct regions as indicated by Grad-CAM does not guarantee that the model truly understands or can generalize from the visualized concepts. Therefore, Grad-CAM should be viewed as one of many tools available for model interpretation, rather than a foolproof solution.

3.4.3 Saliency maps

Saliency maps serve as a visualization technique to detect the components of an input image that significantly influence the output. For what regards the interpretation, bright areas on a saliency map denote image parts with strong influence on the model's prediction. If the saliency map accentuates irrelevant areas, like the background in an object classification task, it may signal the model's misdirection. Moreover, sensitivity to noise or insignificant image sections might suggest overfitting or poor generalization. Saliency maps have practical applications in model debugging and improvement. They enhance model interpretability, explaining complex model behaviors to non-experts and assisting in data cleaning by uncovering biases or mislabeled data through unexpected focus areas. However, limitations exist. Complex images can produce noisy, high-dimensional gradients challenging to interpret directly due to multiple influencing factors. Saliency maps' sensitivity to minor input variations can lead to misleading interpretations, requiring cautious analysis. Additionally, while bright areas signify high influence, they don't clarify the significance of these regions to the model, sometimes making it hard to derive actionable insights without further

context or analysis. Grad-CAM and saliency maps address different needs in model interpretability. Grad-CAM is more suited for understanding the general areas of an image that lead to a model's decision, making it excellent for tasks where the spatial context or specific regions are important. Saliency maps, on the other hand, offer a more granular view, pinpointing exact pixels that influence the output, which can be valuable for fine-grained analysis or when investigating model sensitivity to subtle input variations. The choice between Grad-CAM and saliency maps, or their combination, depends on the specific goals of the interpretability task and the nature of the model and data.

3.4.4 LIME

LIME, standing for Local Interpretable Model-Agnostic Explanations, is a library that aims to explain the predictions made by any classifier or regressor. It achieves this through local approximation using an interpretable model. This is done by altering a single data instance, and monitoring the changes in the prediction outcome. The process involves focusing on a specific prediction to explain, generating a dataset of perturbed samples around that prediction, and analyzing how the model's predictions vary with these sample alterations. Through this analysis, LIME constructs a simpler, locally accurate model, such as a linear model, to approximate the complex model's decision-making around the selected instance. This simpler model sheds light on the critical features that impacted the prediction, enhancing interpretability for that particular scenario. The process begins with LIME selecting an image and generating a series of perturbed versions by segmenting the image into super-pixels and altering them. These perturbed images are then passed through the CNN to observe how modifications affect predictions. LIME employs these variations to train a simple linear model, weighted by the similarity of each perturbed image to the original, aiming to approximate the CNN's complex decision-making process in a localized region. The linear model’s coefficients indicate the influence of each super-pixel on the prediction, providing a quantifiable measure of their importance. The "positive" category contains features that increase the likelihood of a certain prediction. These are highlighted by LIME as having positive weights in the linear model, indicating their significant role in leaning the model's decision towards that particular class. Conversely, the "negative" category includes features that detract from the model making a specific prediction. By distinguishing between these positive and negative influences, LIME provides a nuanced explanation of a CNN's decision-making process. This detailed breakdown helps users and developers not only to trust the model's predictions by understanding the rationale behind them but also to debug and improve the model by identifying misinterpreted features or biases. It enhances interpretability by clarifying which features are decisive for a prediction and which ones may cause the model to hesitate, offering insights into how the model can be refined for better accuracy and reliability.

3.5 Results

3.5.1 Performance Metrics

ResNet101 and DenseNet201 models were both trained on the training dataset described above, according to the following parameter: 25 epochs, Adam optimizer, categorical cross-entropy loss function and a batch size equal to the length of the training dataset divided by 32. The final model weights were the ones of the best performing model in terms of loss (i.e., minimum loss), among all the 25 models trained. The two figures below (Figure 16 and Figure 15) show both loss and accuracy trend for both ResNet101 and DenseNet201 models.

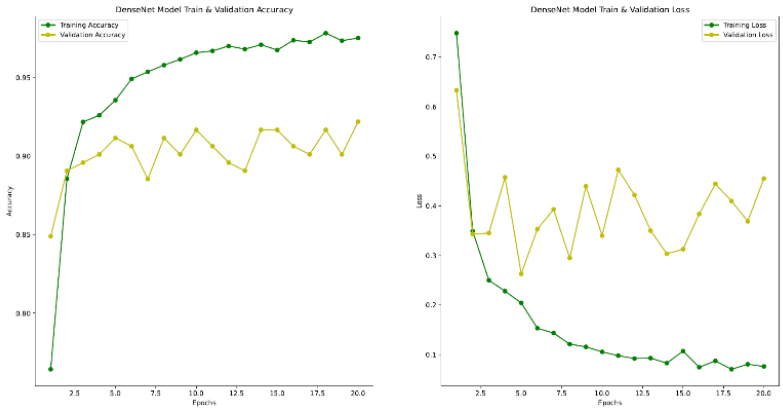

Figure 15: Training statistics for ResNet

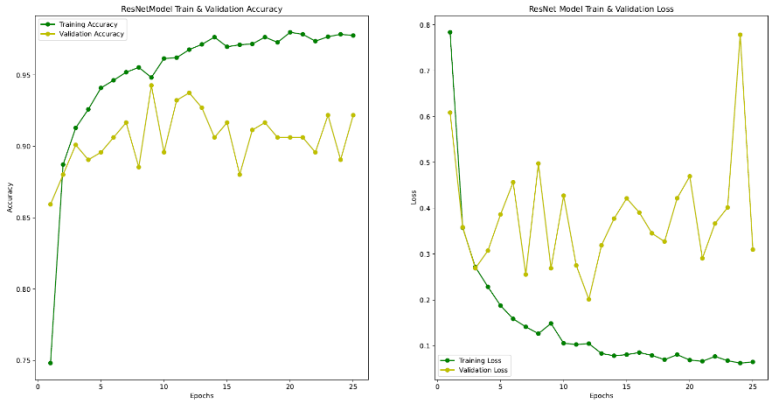

Figure 16: Training statistics for DenseNet

As it is possible to notice from the graph, ResNet model reaches its lowest loss at epoch number 12, even though the highest accuracy on validation set is reached at epoch number 8. For DenseNet model, instead, it reaches lowest loss at epoch number 5 and higher accuracy at epoch number 8. The table below (Table 1) shows the models results, in terms of accuracy and loss, for both train and test datasets.

		Loss	Accuracy
ResNet101	Train	0.0437	0.9905
	Test	0.5128	0.8971
DenseNet201	Train	0.0283	0.9922
	Test	0.4870	0.9107

Table 1: Training Statistics Loss and Accuracy

The classification report presented in the Table 2 offers a comprehensive assessment of the performance of ResNet101v2 model. The model demonstrates a remarkable overall accuracy of 90%, which reflects its effectiveness in classifying scenes across various classes. This accuracy metric is corroborated by both the macro and weighted averages of precision, recall, and the F1-score, each also standing at 90%. These metrics indicate a robust generalization across the classes in the dataset. Precision scores are particularly high for classes such as “beach”, “sea_ice”, and “snowberg”, with scores of 0.99, 0.97, and 0.96, respectively. The recall metric shows exemplary performance in identifying nearly all relevant instances for “desert” and “sea_ice” classes, with recall rates of 0.96 for both. Conversely, the model exhibits some challenges in classifying “river” scenes, as indicated by the lower precision of 0.75 and an F1-score of 0.83. This could potentially be attributed to the inherent difficulty in distinguishing river scenes from other water-related classes or perhaps it is a consequence of the limited representation in the dataset, as suggested by the lower support number for this class. The uniformity of the model's performance across various classes, as reflected in the consistent precision, recall, and F1-scores, suggests a balanced classification capability. Nonetheless, the slightly diminished precision and F1-score for the “river” class may necessitate further investigation. In summary, the model exhibits a strong capability in the classification of complex scenes, yet there remains room for improvement, particularly in classes where the model's precision is less than optimal.

	Precision	Recall	F1-Score	Support
beach	0.99	0.86	0.92	161
desert	0.82	0.96	0.88	120
forest	0.94	0.87	0.9	152
meadow	0.86	0.9	0.88	135
mountain	0.9	0.82	0.86	153
river	0.75	0.94	0.83	112
sea_ice	0.97	0.96	0.96	142
snowberg	0.96	0.94	0.95	144
terrace	0.93	0.9	0.92	144
wetland	0.84	0.85	0.84	137
accuracy			0.9	1400
macro avg	0.9	0.9	0.9	1400
weighted avg	0.9	0.9	0.9	1400

Table 2: ResNet results table

Table 3 delineates the classification report for DenseNet201 model. An overall accuracy of 91% is achieved, signifying a proficient level of correct predictions across the diverse scene classes. This accuracy is reflected uniformly across the macro average and weighted average, both standing at 91%. These averages, which consider the harmonic mean of precision and recall, illustrate the model's balanced performance despite variations in class support. The precision metric reveals that the model exhibits excellent results, with outstanding precision for “sea_ice” at 0.99. Similarly, high precision for the “beach” and “snowberg” classes implies that the model is highly reliable for these scenes.

In terms of recall, the model successfully identifies a majority of the relevant instances across all classes, with “desert”, “sea_ice”, and “terrace” each achieving a recall of above 0.90. The F1-score, which balances precision and recall, remains consistently high across most classes, with “sea_ice” and “terrace” notably achieving 0.96 and 0.95, respectively. Nevertheless, the “wetland” and “river” classes exhibit slightly lower F1-scores of 0.85 and 0.88.

Overall, the DenseNet201 model's classification report attests its effectiveness in scene classification tasks, supported by high precision, recall, and F1-scores, along with an excellent overall accuracy, rendering it a powerful tool for automated scene analysis.

	Precision	Recall	F1-Score	Support
beach	0.96	0.91	0.93	149
desert	0.91	0.91	0.91	139
forest	0.93	0.92	0.92	142
meadow	0.91	0.87	0.89	146
mountain	0.84	0.89	0.86	133
river	0.85	0.92	0.88	129
sea_ice	0.99	0.93	0.96	149
snowberg	0.95	0.92	0.94	144
terrace	0.94	0.96	0.95	138
wetland	0.82	0.88	0.85	131
accuracy			0.91	1400
macro avg	0.91	0.91	0.91	1400
weighted avg	0.91	0.91	0.91	1400

Table 3: DenseNet results table

To sum up, upon comparing the performance metrics of the ResNet101v2 and DenseNet201 models, several key differences can be discerned:

- Overall Accuracy: The DenseNet201 model shows a slight improvement in overall accuracy, with a 91% accuracy compared to the 90% accuracy of the ResNet101v2 model.
- Precision: There are variances in precision for certain classes between the two models. For example, DenseNet201 shows a higher precision for “desert” and “sea_ice” classes, but a slightly lower precision for “mountain” and “river” classes compared to ResNet101v2
- Recall: DenseNet201 exhibits improvements in recall for the “desert”, “forest”, and “terrace” classes. However, the recall for “meadow” and “mountain” scenes is higher in the ResNet101v2 model.
- F1-Score: The DenseNet201 model has higher F1-scores for “beach”, “desert”, “forest”, “sea_ice”, and “terrace” classes. However, the ResNet101v2 model excels with higher F1-scores in “river”, “mountain”, and “meadow” classes
- Class-specific Performance: For certain classes such as “river” and “wetland”, both models show similar challenges, but the extent of the challenge varies. For instance, the “river” class has a higher precision with ResNet101v2 but a higher recall with DenseNet201.

- Consistency Across Classes: While both models show good consistency across classes, the DenseNet201 model appears to have a slight edge in terms of maintaining high precision and recall across most classes, as indicated by the higher macro and weighted averages.

To deep dive deeply into models' performances it would be extremely useful to visualize the two confusion matrixes.

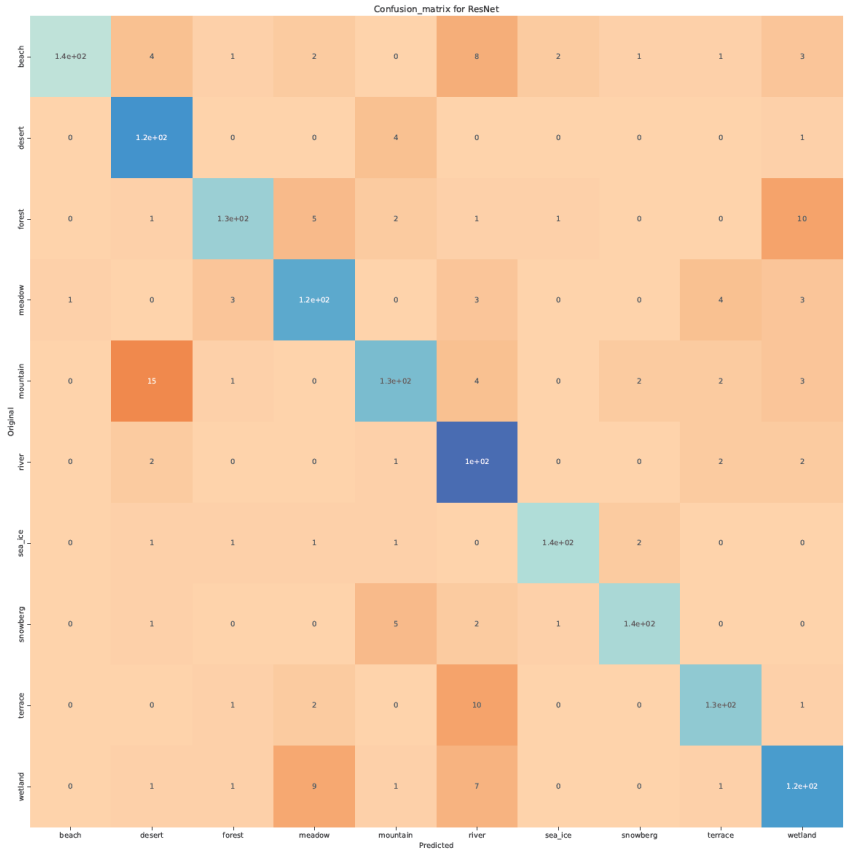

Figure 17: ResNet confusion matrix

Figure 17 and Figure 18 show two confusion matrixes represented as a heatmap for ResNet and DenseNet models' predictions. In this context, the matrix compares the predicted labels (on the x-axis) against the true labels (on the y-axis). The diagonal cells, which represent correct predictions, where the predicted label matches the true label, show as expected higher values, indicated by a lighter blue shade.

For what concerns ResNet (Figure 17), confirming previous table results, classes with higher values in the diagonal are "beach", "snowberg" and "sea ice". Off-diagonal cells are the most interesting ones. They show, indeed, instances where the model has made incorrect predictions. Off diagonal cells which contain greatest values are on the river column. In fact, this class showed the lowest f1

score (i.e., 0.83). Beaches (8 times), terraces (10) and wetlands were the scenarios wrongly classified as rivers. The other two relevant values are: 15 mountains misclassified as deserts (the higher value among all off diagonal cells) and 10 forests mislabeled as wetlands. Both couple of classes (i.e., mountain-desert, wetland-forest) are characterized by a great chromatic similarity which could be the greatest root cause of the errors.

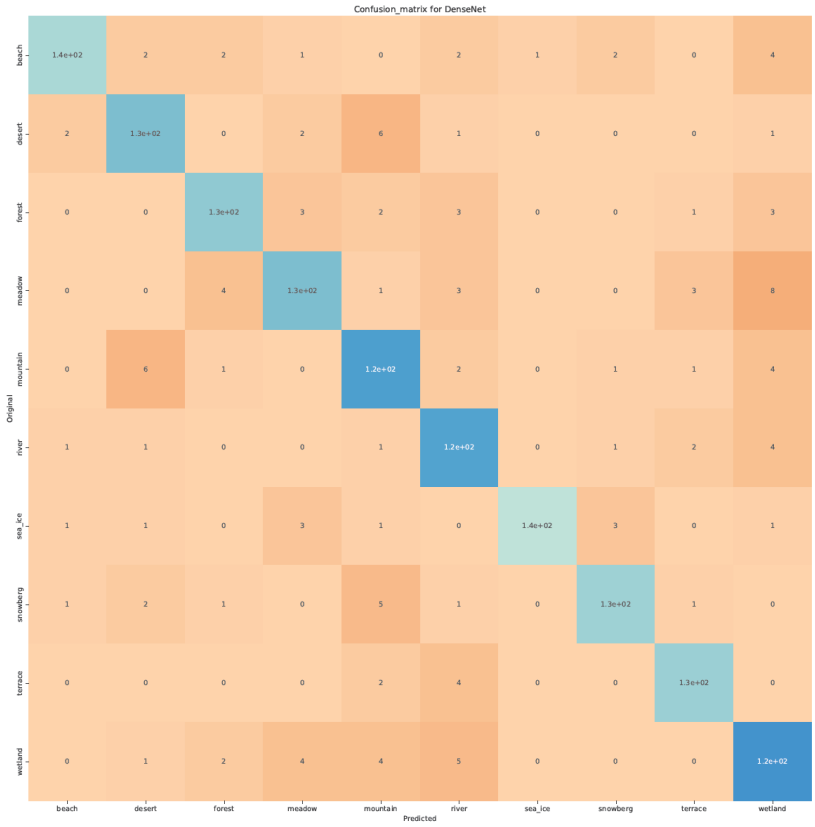

Figure 18: DenseNet confusion matrix

Confusion matrix diagonal, for DenseNet, confirms, instead, that categories with greater numbers of correct predictions are “sea ice”, “snowberg”, “terrace” and “beach”. Looking at the other cells, it is possible to notice that the column with higher number of errors is the “wetland” one. Most of incorrectly labeled images were original from “meadow” class (i.e., 8), but also some pictures from “forest”, “beach”, “mountain” and “river” classes were misclassified (i.e., respectively 4, 3, 4, and 4). Other significant errors come from desert pictures classified as mountains (6) and vice versa (6). The confusion between these classes will be further analyzed in the XAI results section.

The final comment about the confusion matrixes is about the error magnitude. In DenseNet, errors are more equally distributed while in ResNet there are some specific errors that the model make repeatedly.

3.5.2 XAI

3.5.2.1 Filter and Feature Maps

Results analysis, in terms of models explainability, will start from models' filters and features map. This is because, as explained before, they deal with how the models preprocess input images, and with the result of this preprocessing. For this reason, looking at filters' shape and feature maps is the first step to understand ResNet and DenseNet decision making processes. Firstly, first convolutional layer's filters will be visualized.

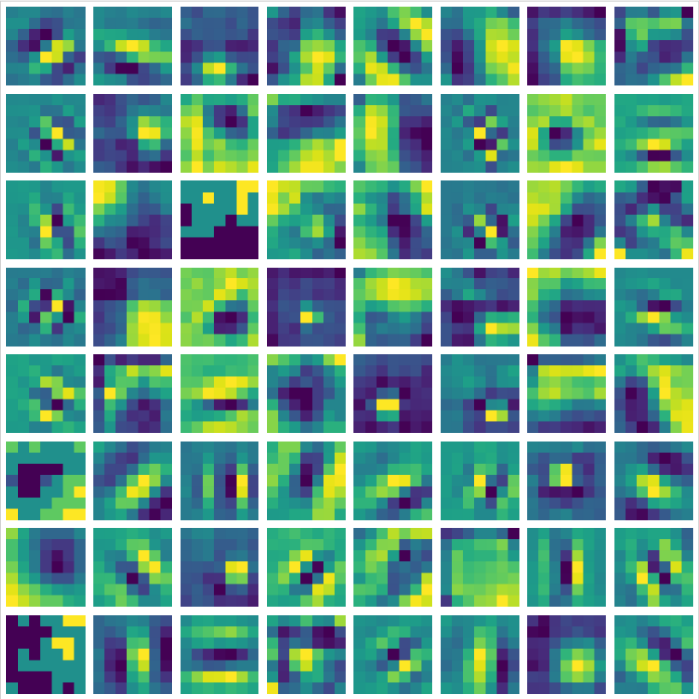

Figure 19: ResNet Filter

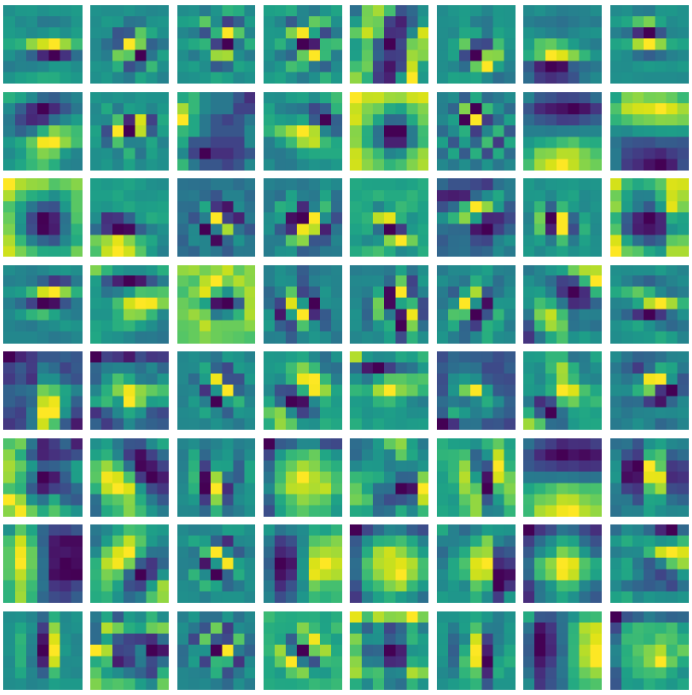

Figure 20: DenseNet Filter

Filters are not input depending and are the same for all images. The filters shape for both models is the following: (7, 7, 3, 64), which means 64 filters of size 7x7 and 3 channels. So, all the first convolutional layer's filters are visualized in the figures above. As you can see, filters are very different between the two models, and therefore also the output, i.e., feature maps, is expected to be very diverse.

The more the layer is in a deeper position, the more complex filters would be, while the visualization would be less interpretable. This is because "deep filters" in both ResNet and DenseNet have shape 1x1. These 1x1 filters are used in deep layers of CNNs, typically for channel-wise transformations, meaning they combine the information across channels to form new features. Since they are 1x1, each filter operates on a single pixel across all channels but independently for each pixel location. The

purpose of these filters is often to reduce dimensionality and to mix information across the channels rather than to detect spatial patterns. A way better interpretable output is provided by feature maps. As explained before, feature maps represent the response of the filters to the input image. So, having a sense of how filters look like thanks to the previous images, now feature maps of different layers will be displayed, to understand better of the images is propagated through the neural network.

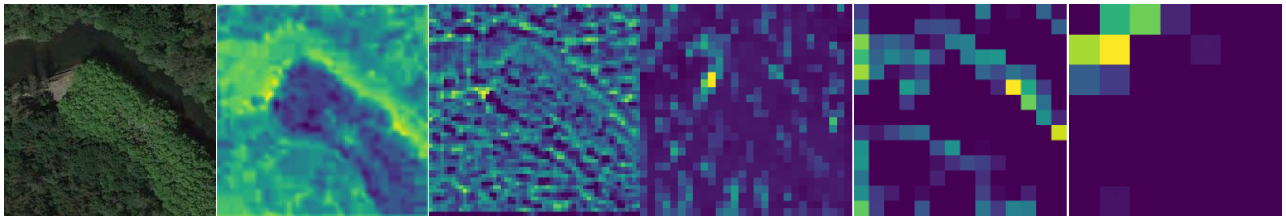

Figure 21: ResNet Feature Maps Series

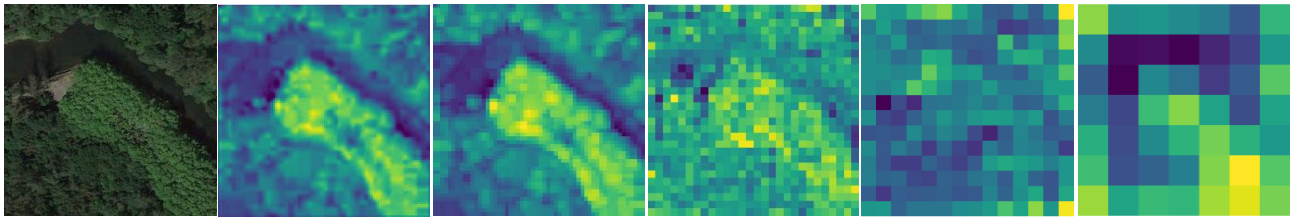

Figure 22: DenseNet Feature Maps Series

The two figures above (Figure 21 and Figure 22) show the same image propagated to 5 different layers in both ResNet and DenseNet. This means that five different filters were applied to that image, with very different outcomes. ResNet in this case focuses on pixel where there is the actual river, while the DenseNet feature maps focus on the land near the river itself. Obviously, this is just a part of a bigger picture, since at each layer a large number of filters is applied (from 64 to 128) and the last picture of the two sequences above, is just a part of the final output of the model processing.

The following two images, in fact, shows the “whole” picture for what concern first and last convolutional layer.

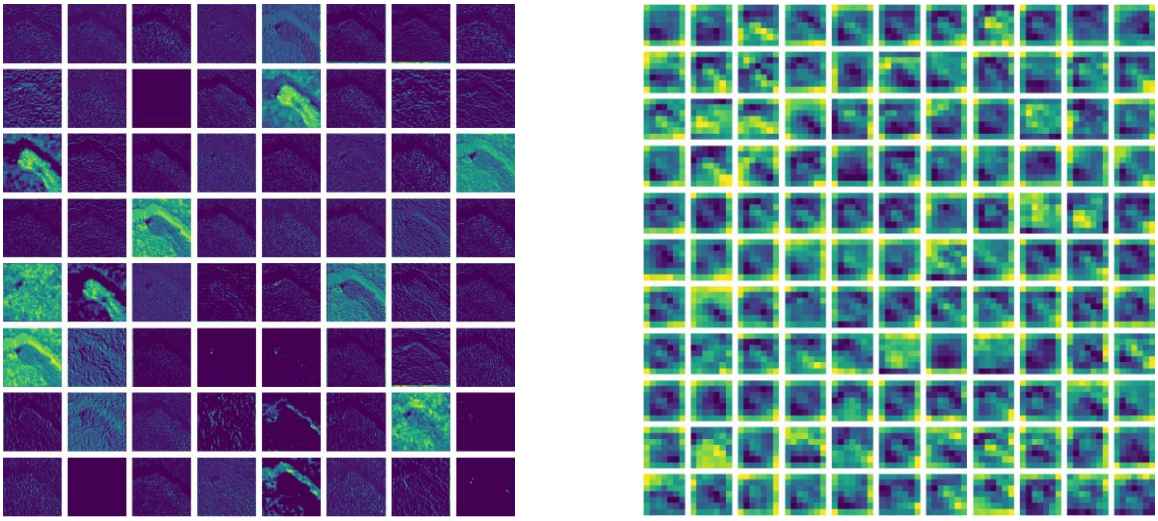

Figure 23: First and Last Layer Feature Maps for DenseNet

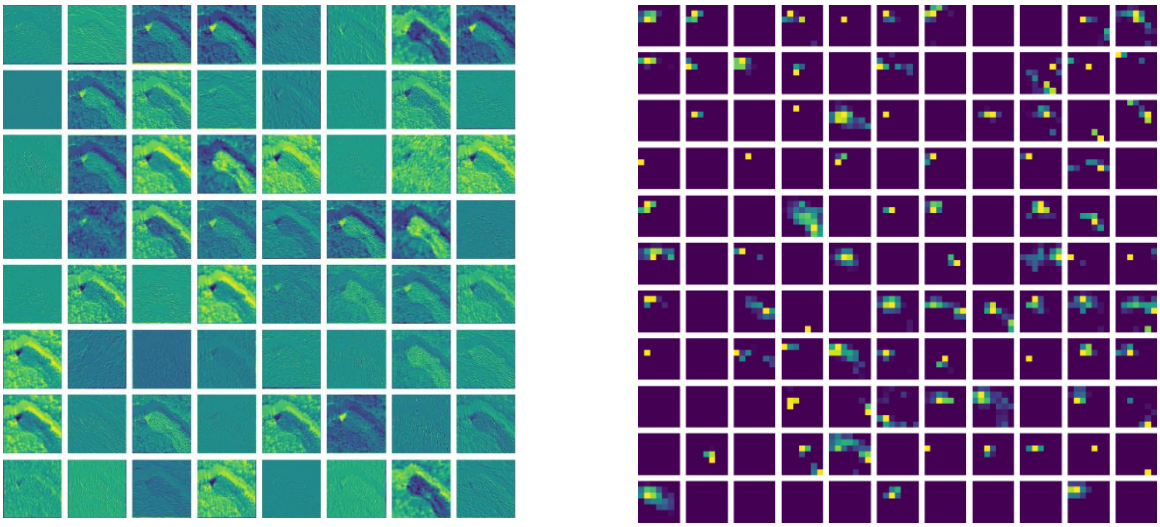

Figure 24: First and Last Layer Feature Maps for ResNet

Also here, it is possible to appreciate a high degree of divergency. It is clear that for ResNet in the first layer lots of feature are activated, while in the last convolutional layer the model focuses on very specific features. DenseNet visualization, instead, offer the opposite scenario. The first layer feature maps are activated by less elements, while the last ones consider the majority of the image.

3.5.2.2 Grad-CAM

After having grasped better all the transformation applied to the input images, it is possible to investigate how all these steps influences the final outcome. In order to do that, the first technique

which serves this purpose, is Grad-Cam. Since it is a technique for visualizing the regions of input that are important for predictions, misclassified images will be further investigated for both models.

In this section are taken into consideration two different images from the test dataset. The selection criterion was the following: for each of the two models, looking at their respective confusion matrixes, were selected two misclassified images from the category with the lowest accuracy, that at the same time the other model was able to label correctly. For what concerns ResNet, it is clear from the confusion matrix (Figure 17) that the highest number of errors comes from mountain categories images classified as deserts. Among these 15 images, the selected image is the one displayed in Figure 25.

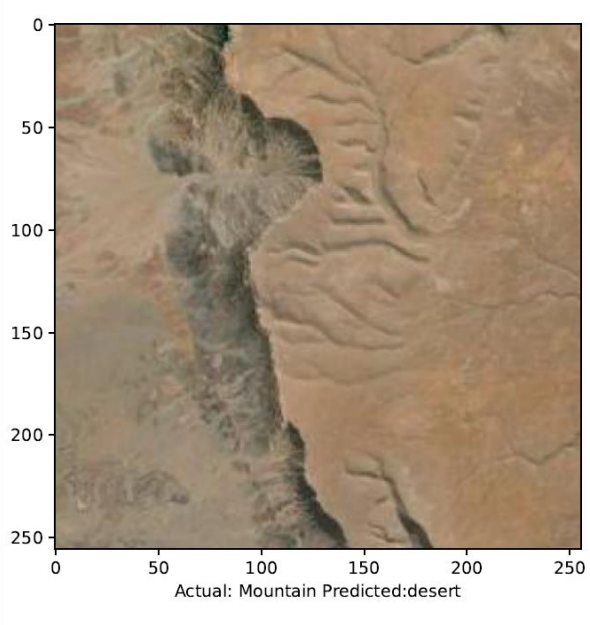

Figure 25: Original image, misclassified by ResNet, for Grad-CAM

As said before, DenseNet model classified it correctly. The following Figure 26, instead represents the Grad-CAM of both models applied to the image. Usually, as aforementioned, all parts highlighted in red correspond to picture's zone which gave greater contribution to the output. How this contribution affects effectively the classification is controversial so it is better to focus on the zone highlighted by different colors rather than paying attention just to the red ones. Starting from similarities between the two heatmaps, it is very evident that both models perceive at the same way the bluish part, which corresponds to the mountain part which stands out from the rest. Indeed, what really differs it the rest of the mountain. In ResNet model, this is perceived more or less such as the peripheral part of the picture, while, in DenseNet one, a pattern is detected and highlighted as a yellowish region. This is the reason why on model (i.e., ResNet) classifies it as desert, and the

other one (i.e., DenseNet), instead, classifies it as mountain. From the human eye perspective, it can be hard to distinguish between desert and mountain, especially from a chromatic point of view. So, while it is understandable the misclassification, at same time it is impressive that DenseNet implementation is able to understand that the scene proposed in this case is a mountain.

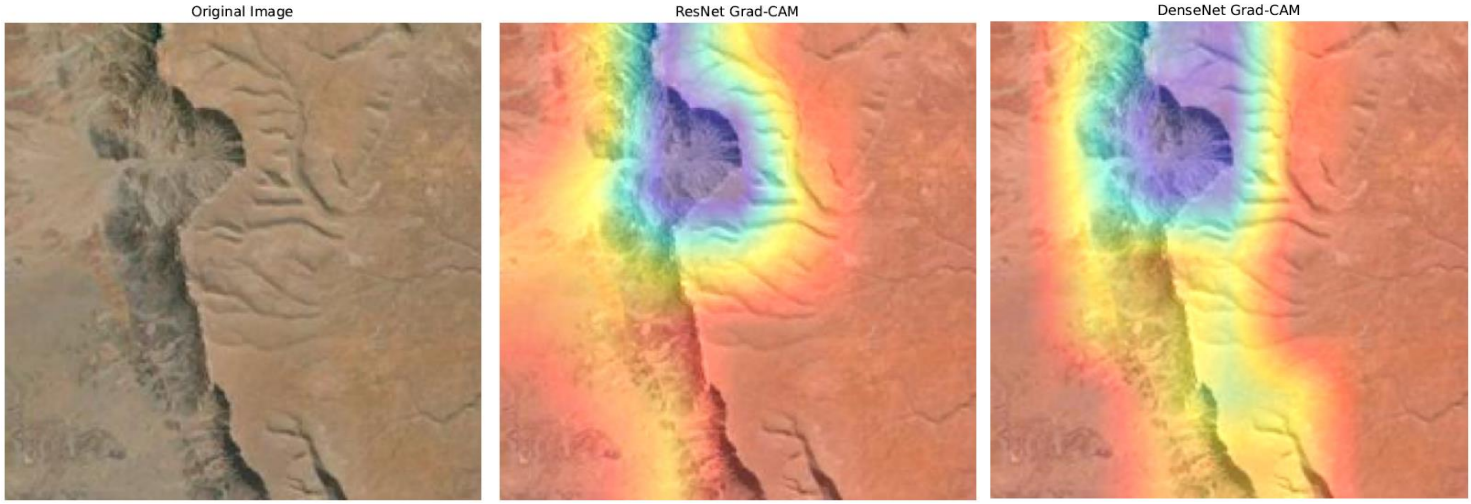

Figure 26: On the left we have the original image, then the ResNet Grad-CAM and after the DenseNet Grad-CAM

The DenseNet model confusion matrix (Figure 18), instead, suggests as the most misinterpreted category the terrace one. In fact, 11 out 140 terrace images were classified as river. Therefore, the image below (Figure 27) was selected because DenseNet classified it incorrectly as a river, while ResNet was able to understand that it was actually terrace.

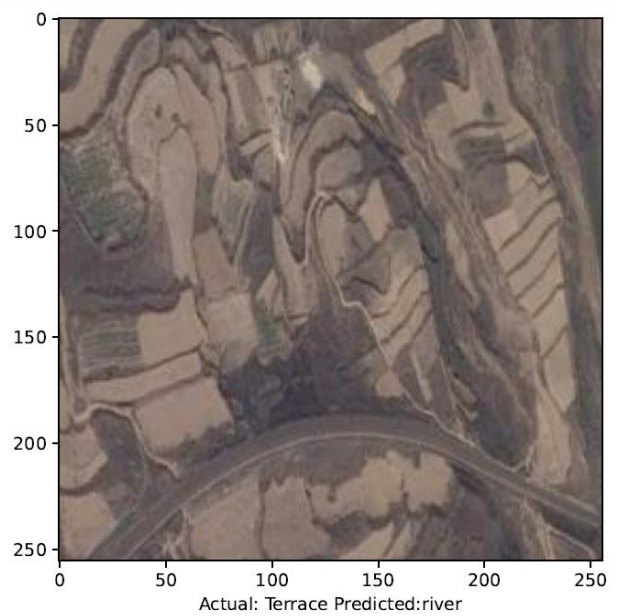

Figure 27: Original image misclassified by DenseNet

As before, the following Figure 28, instead represents the Grad-CAM of both models applied to the image. Here there are less similarities among the two heatmaps, with respect to the previous situation. The bluish region matches partially between the two grad-cams only in bottom left corner. There is agreement, for what concern red regions, in the upper part of the image and in the zone immediately above the bluish region on the bottom. Analyzing the differences, the most significant part regards the big bluish region that is present in DenseNet Grad-CAM, which is totally absent in the ResNet one. In the bottom part there is some sort of disagreement too. In fact, bluish and yellowish regions do not come up in ResNet, while there is a strong presence for DenseNet. An interpretation that could lead to explain the misclassification of this picture consists in the fact that in DenseNet Grad-CAM it is highlighted a “sinusoidal” pattern (i.e., the red part between the two bluish regions), which can vaguely remember the shape of a river. In contrast with the previous use case, the misclassification roots can be found in the shapes more than in the colors.

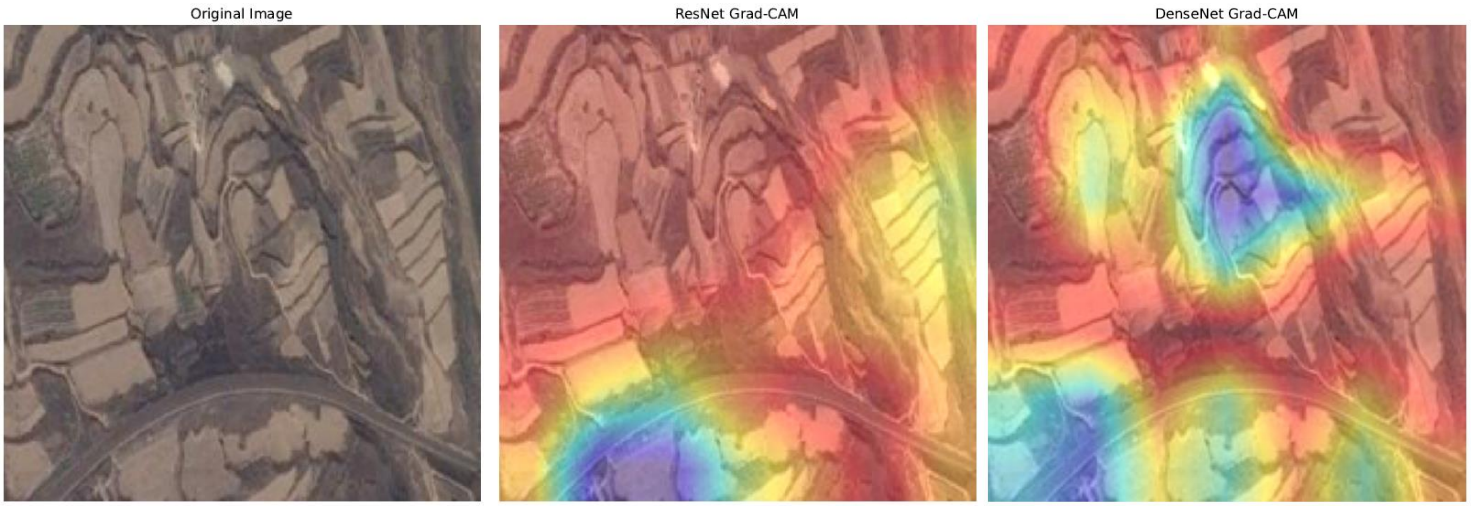

Figure 28: On the left we have the original image, then the ResNet Grad-CAM and after the DenseNet Grad-CAM

3.5.2.3 Saliency Maps

Saliency maps visualization offers a slightly different point of view regarding models' interpretability. Since they offer a pixel-wise approach, the granularity is not completely suited for a scene recognition task. Despite that, for sake of completeness, integrated gradient saliency maps will be visualized too. The image that will be considered in this section is the terrace scenario image from the paragraph above, in Figure 28. It is useful to recall that it was classified as river by DenseNet, and, correctly, as terrace by ResNet. The visualization is composed by three distinct images:

- A normalized saliency map derived from the computed gradients (Left image Figure 29 and Figure 30).
- A normalized version of the original input image, like the one fed to the algorithm (Central image Figure 29 and Figure 30).
- A composite image blending the saliency map and the input image to visualize the most influential regions in the context of the actual image (Right image Figure 29 and Figure 30).

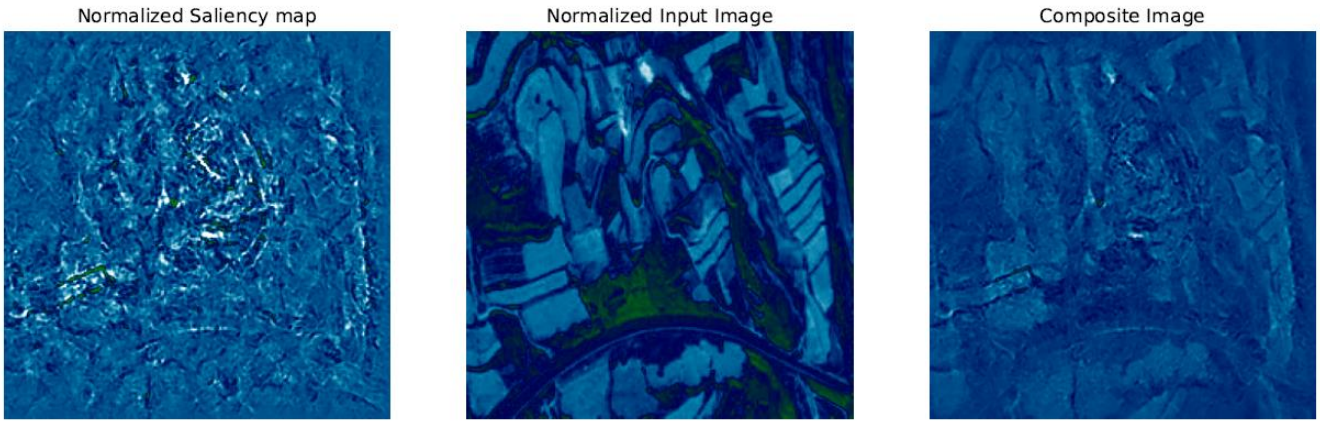

Figure 29: ResNet Integrated Gradient Saliency Map

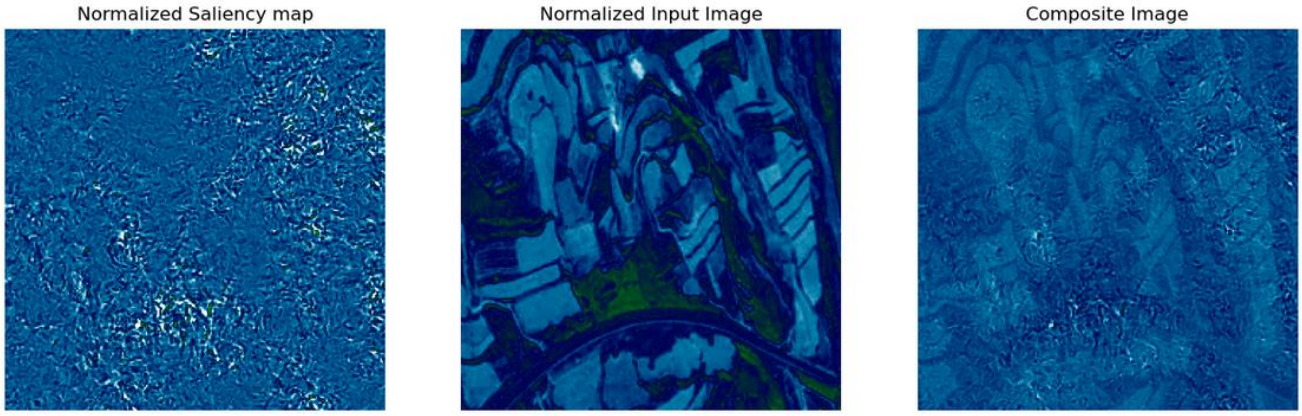

Figure 30: DenseNet Integrated Gradient Saliency Map

Comparing the two images on the left (i.e., the first images of each row), it is evident that activated pixels focus on very different areas. For what regards ResNet, brighter pixels are concentrated in the center of the image. In DenseNet saliency maps, instead, the greatest concentration of white pixels is in the bottom part of the picture, where there is an object with the shape similar to a river. As discussed in the Grad-CAM section, DenseNet prediction seems to focus mainly on this misleading region, while neglecting the rest of the scene, which is indeed characterized by edges, shapes and patterns typical of the terrace class.

3.5.2.4 LIME

The last XAI techniques that will be discussed is produced by LIME python library. As explained above, interpreting LIME output is very straightforward, since it highlights picture's portions which influence the most the output. For this final analysis, it will be considered an image that was classified

correctly by both models (Figure 31). Furthermore, only ResNet output will be considered in this section.

It is an image which belongs to the beach category, and to human eye test, it does not present any misleading pattern or object. The main aim of this visualization is to better understand the overall decision process when predicting the correct class.

Figure 31: Input image for LIME library

The image on the left of Figure 32 puts in evidence only the regions which contributes positively to outcome. “Positive" here means that the presence or the characteristics of these super-pixels increase the model's confidence in its prediction for the given class. It seems reasonable that the model considers as most relevant all the coastal regions and a part of the sea. On the right, instead, the picture shows at the same time regions which have a positive contribution, in green, and the ones which contribute negatively. In this case, "negative" influence means that these areas of the image, if altered or removed, could potentially increase the model's confidence in the prediction or possibly lead to a different prediction altogether. This can be particularly illuminating when you're trying to understand what aspects of the input might be confusing the model or leading it towards incorrect predictions. It is clear here that what can confuse the model is the sea foam, which, if taken separately from the rest of the image, can be difficult to classify. Given that, the “positive” parts succeeded in helping the model to label the picture correctly.

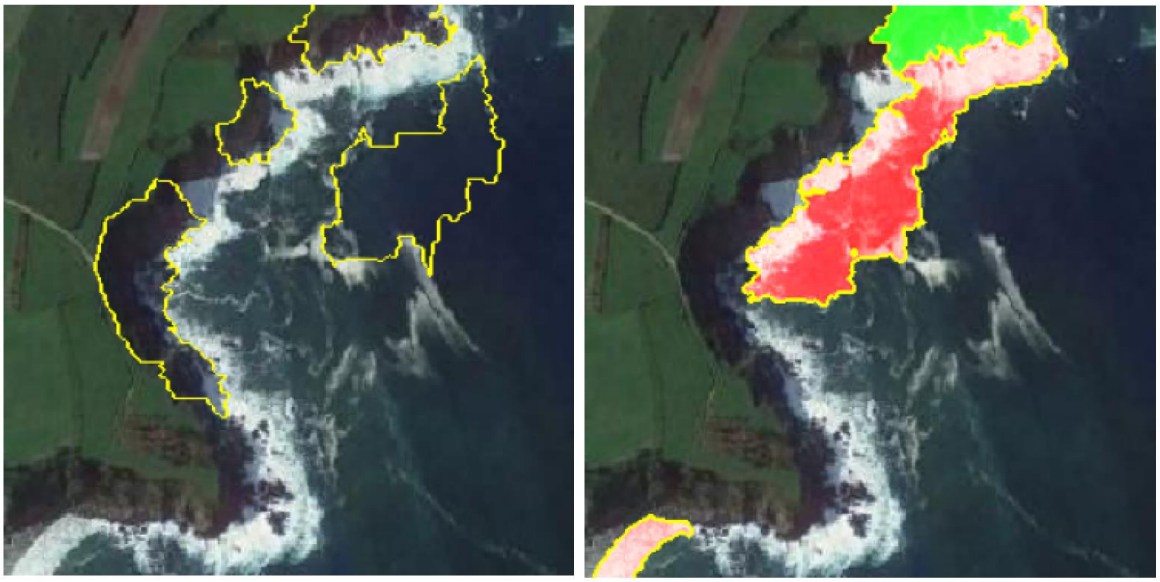

Figure 32: LIME output

3.5.3 Conclusions about Models' Performances

To sum up, ResNet and DenseNet models offers very comparable performances in terms of accuracy, precision and recall, with DenseNet achieving slightly more remarkable results. Since it is difficult to choose one model over the other just considering these metrics, also XAI dimension is taken into considerations. Since the degree of interpretability is the same due to the similar architectures, explainability would be investigated to assess the quality of the decision-making process.

The contribution of each technique can be broken down as follows: filters and features map visualization serve users purposes to understand transformation applied to input data, and its propagation through the layers; Grad-CAM, Saliency Maps and LIME deep dive into the actual decisional process.

From the output analysis, ResNet demonstrates a higher quality for what concerns image understanding and generalization with respect to DenseNet. In conclusion, with an appropriate data augmentation aimed to solve current gaps in the dataset, accuracy metrics and XAI techniques interpretation suggests ResNet as the more effective classification model.

4. Conclusions

4.1 Summary of Findings and Contributions

The findings of this thesis underscore the critical role of XAI in enhancing the transparency and accountability of CNNs, through a practical application in the domain of aerial image scene classification.

By systematically applying XAI techniques to state-of-art CNN models, such as ResNet and DenseNet, the research demonstrates that it is possible to achieve a robust balance between model interpretability and performance. The practical application involving the NWPU-RESISC45 dataset, refined to focus on environmentally significant classes, reveals that some modification and adaptations to XAI techniques do not necessarily come at the cost of accuracy. This aspect supports the initial hypothesis which argued that models' enhanced understandability does not implies more inaccurate model. In fact, both ResNet and DenseNet reached remarkable degree in terms of both accuracy (i.e, respectively 0.9 and 0.91) and understandability.

The second pivotal insight is that XAI not only makes the decision-making processes of these complex models more transparent but also aids in identifying and mitigating biases, thereby enhancing overall model reliability, trustworthiness and performances. This finding addresses the widespread prejudice which implies that XAI techniques serves external purpose, rather than internal ones. For instance, in the context of the practical implementation of ResNet and DenseNet, the class-wise analysis of Grad-CAM and Saliency maps output is essential in diagnosing limitation in the dataset's images, providing essential aid to model's designer.

Furthermore, the comparative analysis between ResNet and DenseNet highlights the structural nuances that contribute to their performance and interpretability. ResNet's residual connections and DenseNet's dense connectivity both offer unique advantages that can be leveraged depending on the specific requirements of the task at hand. This dual-model approach enriches the findings by offering diverse perspectives on the effectiveness of XAI techniques across different neural network architectures. Indeed, when two models are so similar regarding accuracy-related metrics, such ResNet and DenseNet in this specific case, the possibility to explore an additional dimension represented by the understandability landscape, can be considered game-changing. In the future, there will be no ML/AI application deployed without considering its efficiency from XAI perspective.

4.2 Limitations and Challenges

Despite the significant contributions and insights provided by this thesis, there are several limitations and challenges that need to be considered. In fact, considering the hypothesis, confirmed in the CNNs application, that enhancing interpretability does not necessarily compromise performance, it must be stressed that this balance can be highly context-dependent. In some scenarios, increasing interpretability might still lead to a decrease in accuracy. Indeed, thesis findings hold just in the field of CNNs, while it is impossible to generalize this conclusion for all AI models.

Another critical aspect, considered out of scope within this thesis regards, concerns computational expense. In fact, the XAI techniques applied, such as Grad-CAM and Saliency Maps, can be computationally intensive. This complexity might pose challenges, for instance, for real-time applications, which represent one of the major sectors of deployment for CNNs. Future research could explore more efficient XAI techniques or optimizations of existing ones.

Finally, the last point which limits the scope of the whole dissertation is the lack of interdisciplinary collaboration. The thesis emphasizes the need for explanations that are understandable to both domain experts and non-experts, while proposing itself as a first tentative guide. However, the actual human-centric evaluation of these explanations, such as user studies or cognitive assessments, is not deeply explored. Understanding how different users interpret and perceive these outputs is crucial for developing better XAI techniques and assess model understandability.

4.3 Recommendations for Future Research

Addressing these limitations and challenges can pave the way for further advancements in the field of Explainable Artificial Intelligence.

In order to overcome generalizability issues, an exhaustive analysis about all model-tailored XAI techniques should be conducted. Furthermore, all state-of-art models performances should be re-assed considering understandability criteria.

Lack of interdisciplinary perspective, indeed, can be tackled by collaborating with experts from psychology, human-computer interaction, ethics, and law. This would provide valuable plurality of points of view, that would enhance development and evaluation of XAI techniques. This interdisciplinary approach can help developing advanced visualization tools that provide more intuitive and interactive ways to explore model decisions, increasing remarkably XAI techniques' effectiveness. These tools should be beneficial to both experts and non-experts, making complex model behaviors more accessible and understandable.

References

(http://deepglobe.org). (n.d.).

Alejandro Barredo Arrieta, N. D.-R.-L. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 82-115. doi:https://doi.org/10.1016/j.inffus.2019.12.012

Alex Krizhevsky, I. S. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097–1105. doi:10.5555/2999134.2999257

Alexey Dosovitskiy, L. B. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. Retrieved from https://arxiv.org/abs/2010.11929

Arun Nair, P. S. (2015). Massively parallel methods for deep reinforcement learning. Retrieved from https://arxiv.org/abs/1507.04296

Ashish Vaswani, N. S. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*, (pp. 2481-2495). doi:10.1109/TPAMI.2016.2644615

Chattopadhay, A. (2018). Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, (pp. 839-847). doi:doi: 10.1109/WACV.2018.00097

Cheng, G., Han, J., & Lu, X. (2017). Remote Sensing Image Scene Classification: Benchmark and State of the Art. *Proceedings of the IEEE*, (pp. 1865-1883). doi:10.1109/JPROC.2017.2675998

D. E. Rumelhart, G. E. (1986). Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition*, 318-362. doi:https://dl.acm.org/doi/10.5555/104279.104293

Dan C. Ciresan, U. M. (2011). Flexible, High Performance Convolutional. *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, (pp. 1237-1242). doi:10.1109/ICDAR.2011.229

Daniel Smilkov, N. T. (2017). SmoothGrad: removing noise by adding noise. Retrieved from https://arxiv.org/abs/1706.03825

Divyani Kohli, R. S. (2012). An ontology of slums for image-based classification. *Computers, Environment and Urban Systems*, 154-163. doi:https://doi.org/10.1016/j.compenvurbsys.2011.11.001.

Doran, D. &. (2017). What Does Explainable AI Really Mean? A New Conceptualization of Perspectives.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 193–202. doi:https://doi.org/10.1007/BF00344251

Gao Huang, Z. L. (2017). Densely Connected Convolutional Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 4700-4708). Retrieved from https://openaccess.thecvf.com/content_cvpr_2017/html/Huang_Densely_Connected_Convolutional_CVPR_2017_paper.html

Geoff Pleiss, D. C. (2017). Memory-Efficient Implementation of DenseNets. Retrieved from arXiv preprint arXiv:1707.06990

Ghimire, B. R. (2012). An evaluation of bagging, boosting, and random forests for land-cover classification in Cape Cod, Massachusetts, USA. *GIScience & Remote Sensing*, 623–643.

Goodfellow Ian, Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. Retrieved from https://books.google.it/books?hl=it&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=Bengio+et+al.,+2016&ots=MON4gnpAPY&sig=s98RQ2C_k6xoogGh1EL9oA_WH7c#v=onepage&q=Bengio%20et%20al.%2C%202016&f=false

Graham, B. (2014). Fractional max-pooling. Retrieved from https://arxiv.org/abs/1412.6071

Hansen, M. C. (2013). High-Resolution Global Maps of 21st-Century Forest Cover Change. *Science*, 850-853. Retrieved from https://www.science.org/doi/abs/10.1126/science.1244693

Helber, P., Bischke, B., Dengel, A., & Borth, D. (2017). EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, (pp. 2217-2226). doi:10.1109/JSTARS.2019.2918242

Ian Goodfellow, J. P.-A.-F. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

Ibrahim, R. a. (2023). Explainable convolutional neural networks: A taxonomy, review, and future directions. *ACM Computing Surveys*, 1-37. doi:https://doi.org/10.1145/3563691

Jifeng Dai, Y. L. (2016). R-FCN: Object Detection via Region-based Fully Convolutional Networks. *Advances in Neural Information Processing Systems 29 (NIPS 2016)* .

K Simonyan, A. V. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. Retrieved from https://arxiv.org/abs/1312.6034

Kaiming He, X. Z. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 770-778). Retrieved from https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

Kavukcuoglu, K., Ranzato, M., Fergus, R., & LeCun, Y. (2009). Learning invariant features through topographic filter maps. *IEEE conference on computer vision and pattern recognition.* doi:10.1109/CVPR.2009.5206545

LeCun, Y. Y. (2015). Deep learning. *Nature*, 436-444. doi:10.1038/nature14539

Matthew D. Zeiler, R. F. (2013). Stochastic Pooling for Regularization of Deep Convolutional Neural Networks. Retrieved from https://arxiv.org/abs/1301.3557

McCulloch, W. a. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 115-133. doi:http://dx.doi.org/10.1007/BF02478259

Melgani, F., & Bruzzone, L. (2004). Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on Geoscience and Remote Sensing*, (pp. 1778-1790). doi:10.1109/TGRS.2004.831865

Min Lin, Q. C. (2013). Network in network. Retrieved from https://arxiv.org/abs/1312.4400

Mingxing Tan, Q. L. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *Proceedings of the 36th International Conference on Machine Learning*, (pp. 6105-6114). Retrieved from http://proceedings.mlr.press/v97/tan19a.html?ref=jina-ai-gmbh.ghost.io

Mukund Sundararajan, A. T. (2017). Axiomatic attribution for deep networks. *Proceedings of the 34th International Conference on Machine Learning*, (pp. 3319-3328). Retrieved from http://proceedings.mlr.press/v70/sundararajan17a/sundararajan17a.pdf

Mulla, D. (2013). Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps. *Biosystems engineering*, 358-371. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S1537511012001419

Newsam, Y. Y. (2010). Bag-of-visual-words and spatial extensions for land-use classification. *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, (pp. 270–279). doi:https://doi.org/10.1145/1869790.1869829

Olga Russakovsky, J. D.-F. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 211-252. doi:https://doi.org/10.1007/s11263-015-0816-y

O'shea, K. a. (2015). An introduction to convolutional neural networks. *Arxiv*. Retrieved from https://arxiv.org/abs/1511.08458v2

Raina, R. A. (2009). Large-scale deep unsupervised learning using graphics processors. *Proceedings of the 26th annual international conference on machine learning*, (pp. 873-880). doi:10.1145/1553374.1553486

Ramprasaath R. Selvaraju, M. C. (2017). Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, (pp. 618-626). Retrieved from https://openaccess.thecvf.com/content_iccv_2017/html/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.html

Rawat, W., & Wang, Z. (2017). Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *https://direct.mit.edu/neco/article-abstract/29/9/2352/8292/Deep-Convolutional-Neural-Networks-for-Image?redirectedFrom=fulltext*, 2352–2449. doi:https://doi.org/10.1162/neco_a_00990

Riccardo Guidotti, A. M. (2019). A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv. 51*, 42. doi:https://doi.org/10.1145/3236009

Rippel, O., Snoek, J., & Adams, R. P. (2015). Spectral Representations for Convolutional Neural Networks. In C. C. Garnett, *Advances in Neural Information Processing Systems.* Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2015/file/536a76f94cf7535158f66cfbd4b113b6-Paper.pdf

Ronneberger, O. F. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention* . doi:https://doi.org/10.1007/978-3-319-24574-4_28

Sainath, T. N., Mohamed, A.-r., Kingsbury, B., & Ramabhadran, B. (2013). Deep convolutional neural networks for LVCSR. *IEEE International Conference on Acoustics, Speech and Signal Processing.* doi:10.1109/ICASSP.2013.6639347

Saining Xie, R. G. (2017). Aggregated Residual Transformations for Deep Neural Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1492-1500). Retrieved from https://openaccess.thecvf.com/content_cvpr_2017/html/Xie_Aggregated_Residual_Transformations_CVPR_2017_paper.html

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 85-117. doi:https://doi.org/10.1016/j.neunet.2014.09.003.

Sermanet, P., Chintala, S., & LeCun, Y. (2012). Convolutional neural networks applied to house numbers digit classification. *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, (pp. 3288-3291). Retrieved from https://ieeexplore.ieee.org/abstract/document/6460867

Simon Jegou, M. D. (2017). The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, (pp. 11-19). Retrieved from https://openaccess.thecvf.com/content_cvpr_2017_workshops/w13/html/Jegou_The_One_Hundred_CVPR_2017_paper.html

Simonyan, K. a. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*. Retrieved from https://arxiv.org/abs/1409.1556

Sutskever, K. A., Hinton, I. a., & E, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems.* Retrieved from https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Szegedy, C. a. (2015). Going deeper with convolutions., (p. Proceedings of the IEEE conference on computer vision and pattern recognition). doi:10.1109/CVPR.2015.7298594

Tang, Y. (2013). Deep learning using linear support vector machines. Retrieved from https://arxiv.org/abs/1306.0239

Tobias Pohlen, A. H. (2017). Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 4151-4160). Retrieved from https://openaccess.thecvf.com/content_cvpr_2017/html/Pohlen_Full-Resolution_Residual_Networks_CVPR_2017_paper.html

Werbos, P. (1981). Generalization of Backpropagation with Application to a Recurrent Gas Market Model. *Neural Networks*, 339-356. doi:https://doi.org/10.1016/0893-6080(88)90007-X.

Wiesel., & Hubel. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 574–591. doi:10.1113/jphysiol.1959.sp006308

Xia, G.-S., Hu, J., Hu, F., Shi, B., Bai, X., Zhong, Y., . . . Lu, X. (2017). AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification. *IEEE Transactions on Geoscience and Remote Sensing*, (pp. 3965-3981). doi:10.1109/TGRS.2017.2685945

Xu, J., & Fortes, J. A. (2010). Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments., (pp. 179-188). doi: 10.1109/GreenCom-CPSCom.2010.137.

Xu, Z. a. (2015). A discriminative CNN video representation for event detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 1798--1807). doi:10.1109/CVPR.2015.7298789

Xuelin Qian, Y. F.-G. (2018). Pose-Normalized Image Generation for Person Re-identification. *Proceedings of the European Conference on Computer Vision (ECCV)*, (pp. 650-667). Retrieved from https://openaccess.thecvf.com/content_ECCV_2018/html/Xuelin_Qian_Pose-Normalized_Image_Generation_ECCV_2018_paper.html

Yann LeCun, L. B. (1998). GradientBased Learning Applied to Document. *Proceedings of the IEEE*, 2278-2324. doi:10.1109/5.726791

Yu, D. W. (2014). Mixed Pooling for Convolutional Neural Networks. *Rough Sets and Knowledge Technology: 9th International Conference*, (pp. 364-375). doi:https://doi.org/10.1007/978-3-319-11740-9_34

Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. Retrieved from arXiv preprint arXiv:1605.07146

Zeiler, M. F. (2014). Visualizing and Understanding Convolutional Networks. *Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science*. doi:https://doi.org/10.1007/978-3-319-10590-1_53