

Titre: Classification of Methane Satellite Image Retrievals
Title:

Auteur: Mohamed Yassir El Aoufir
Author:

Date: 2023

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: El Aoufir, M. Y. (2023). Classification of Methane Satellite Image Retrievals
Citation: [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie.
<https://publications.polymtl.ca/57011/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/57011/>
PolyPublie URL:

**Directeurs de
recherche:** Ettore Merlo
Advisors:

Programme: Génie informatique
Program:

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Classification of Methane Satellite Image Retrievals

MOHAMED YASSIR EL AOUFIR

Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie informatique

Décembre 2023

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Classification of Methane Satellite Image Retrievals

présenté par **Mohamed Yassir EL AOUFIR**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Heng LI, président

Ettore MERLO, membre et directeur de recherche

Quentin CAPPART, membre

DEDICATION

*I dedicate my work to my family, father, mother, brother, sister, grandparents, uncles,
aunts, and friends.*

ACKNOWLEDGEMENTS

I would like to thank several people who allowed me to accomplish this work under the best conditions.

First, I would like to thank my professor and research director, Ettore Merlo, who instructed, guided, and advised me throughout the entirety of my work, and my lab partner and now friend, Zhenyu Yang, who was extremely helpful, supportive, and instructive. I really enjoyed working with you both.

Second, I would like to thank the GHGSAT team which made this work possible and which was very helpful in responding and providing timely feedback on the various questions and concerns we may have had.

Third, I would like to thank the jury members, Professor Heng Li and Professor Quentin Cappart, for taking the time to review my work.

Fourth, I would like to thank my parents who always supported me and allowed me to study in Canada, as well as my brother, sister, and friends who cheered me up in times of doubt.

RÉSUMÉ

Le combat contre le changement et réchauffement climatique est un enjeu majeur au 21^e siècle. Le méthane est le deuxième gaz à effet de serre d'origine humaine le plus émis et l'augmentation continue de ses niveaux d'émissions présente désormais une préoccupation internationale. En 2015, l'Accord de Paris a été adopté par 196 pays dans lequel des efforts coordonnés pour réduire les taux d'émission ont été mis en place.

Les différentes limitations présentées par les méthodes traditionnelles de suivie et d'estimation des émissions ont conduit à l'utilisation croissante de technologies de télédétection telles que les méthodes aéroportées et satellitaires pour suivre les émissions de méthane. À l'exemple de GHGSAT, une société qui exploite des satellites équipés de capteurs hyperspectraux conçus explicitement pour surveiller les émissions de méthane provenant des industries et d'autres sources dans le monde.

Le travail présenté a pour objectif d'implémenter des méthodes d'apprentissage automatique dans le but de pouvoir différencier automatiquement les récupérations des satellites de GHGSAT qui présentent des émissions de méthane de ceux qui n'en présentent pas. Pour ce faire, 16 679 récupérations d'image de méthane qui ont été labellisés par les opérateurs de GHGSAT ont été utilisés.

Des chaines d'image ainsi que des descripteurs d'image ont été conçus après une étude du jeu de données. Ces derniers sont extraits et utilisés pour entraîner et améliorer plusieurs modèles d'apprentissage automatique. L'approche a été rigoureusement évaluée, les meilleurs modèles affichent des précisions (accuracies) de 92% et un taux de détection de 80% pour les récupérations contenant des émissions, atteignant 96% pour les émissions les plus importantes.

ABSTRACT

The 21st century poses an unprecedented challenge to humanity: combating and reducing the negative effects of climate change and global warming caused by greenhouse gases such as carbon dioxide and methane.

Methane is the second most human-induced emitted greenhouse gas, and the continuous rise of its emission levels became an international concern, leading to coordinated efforts to reduce its emissions rates, such as the Paris Agreement, which was adopted by 196 countries in 2015.

Traditional monitoring and emission rate estimation methods present several limitations, leading to the rise of the use of remote sensing technologies such as airborne methods and satellites to track methane emissions. GHGSAT operates satellites equipped with hyperspectral sensors designed explicitly for monitoring methane emissions from industries and other sources worldwide.

The goal is to develop a machine learning model to automatically distinguish GHGSAT retrievals that contain methane emissions from those that do not, using a dataset of 16 779 labeled methane image retrievals from five satellites.

After a study of the dataset, handcrafted channels and features are designed and extracted to train and fine-tune several machine learning models. A rigorous evaluation of this approach demonstrates its validity. The best models attain an accuracy of 92%, a detection rate of retrievals containing emissions of 80%, rising to 96% for the largest emissions, and a precision of 89%. Furthermore, these models achieve an F1 score of 83%, indicating a strong balance between precision and recall in identifying emissions.

TABLE OF CONTENTS

DEDICATION	iii
ACKNOWLEDGEMENTS	iv
RÉSUMÉ	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	xi
LIST OF FIGURES	xii
LIST OF SYMBOLS AND ACRONYMS	xiii
CHAPTER 1 INTRODUCTION	1
1.1 Importance of methane emissions monitoring and detection	1
1.2 Methane emissions monitoring	2
1.2.1 Limits of traditional quantification methods	2
1.2.2 Role of remote imagery	2
1.3 Collaboration and motivation	3
1.4 Scope	4
1.5 Research objective	5
1.6 Contributions	5
1.7 Thesis overview	6
CHAPTER 2 BACKGROUND AND THEORY	7
2.1 Machine learning basics	7
2.1.1 Formal definitions of a machine learning problem	7
2.1.2 Supervised and unsupervised learning	8
2.1.3 Machine learning model	9
2.1.4 Generalization and cross-validation	9
2.2 Classification task	10
2.2.1 Definition	10
2.2.2 Classification metric scores	11

2.2.3	Image classification task	12
2.3	Data preprocessing	13
2.3.1	Feature scaling	13
2.3.2	Feature encoding	14
2.3.3	Dimensionality reduction	14
2.3.4	Image noise reduction	15
2.4	Parametric Models	15
2.4.1	Linear Model	15
2.4.2	Logistic regression	16
2.4.3	Loss and cost function	17
2.4.4	Gradient Descent	17
2.5	Neural Networks and deep learning	18
2.5.1	Architecture	18
2.5.2	Learning process	20
2.5.3	Activation functions	21
2.6	Convolutional Neural networks	22
2.6.1	Convolution layer	22
2.6.2	Pooling layer	23
2.6.3	Architecture	24
2.7	Decision Tree	24
2.8	Ensemble methods	25
2.8.1	Bagging	26
2.8.2	Boosting	27
2.8.3	Stacking	28
2.9	Hyperparameters and model selection	28
2.9.1	Hyperparameters	28
2.9.2	Model selection	29
CHAPTER 3	Litterature review	30
3.1	Computer vision	30
3.2	Image classification and related tasks	30
3.2.1	Pascal VOC and ImageNet Large scale visual recognition challenge	31
3.2.2	Traditional feature extraction approaches	31
3.2.3	Automatic feature extraction approaches	33
3.2.4	Takeaway	35
3.3	Introduction to hyperspectral data classification	35

3.4	Automatic smoke detection	36
3.4.1	Traditional approaches	36
3.4.2	Deep learning approaches	38
3.4.3	Discussion	39
3.5	Automatic methane plume detection	40
3.5.1	Methane leak detection in infrared camera	40
3.5.2	Airborne and satellite retrievals	43
3.5.3	Discussion	45
CHAPTER 4	METHODOLOGY	47
4.1	Introduction and justification of the approach	47
4.1.1	Overview of the approach	47
4.1.2	Justification	48
4.2	Dataset analysis	49
4.2.1	Dataset presentation	49
4.2.2	Visual aspect and observations	51
4.2.3	Image size and cropping	54
4.2.4	Class balance and plumes	56
4.2.5	Satellites exploration	56
4.2.6	Emission rates analysis and categorization	58
4.2.7	Geolocation study	59
4.2.8	CH ₄ plume pixels values and proportions	63
4.2.9	Correlation of channels pixel values	64
4.3	Channels creation	65
4.3.1	Intensity transformed channels	65
4.3.2	Gaussian blurred channels	66
4.3.3	Filtered intensity transformed channels	66
4.4	HandCrafted features	66
4.4.1	Retrieval dimension features and satellite ID	67
4.4.2	Histograms of intensities	67
4.4.3	Relative statistics	68
4.4.4	Histogram of oriented gradients (HOG)	69
4.4.5	Intensity concentration features	69
4.4.6	Channel correlations	71
4.4.7	Geospatial features	72
4.5	Features selection and models fine-tuning	73

4.5.1	Features, scaling, and dimensionality reduction assessments	73
4.5.2	Models fine-tuning	74
4.6	Simple informative comparison: Image-based features versus CNNs	75
4.6.1	Dataset	75
4.6.2	Experiment	76
4.7	Approach assessment	76
4.8	Detection pipeline	77
4.9	Environment and libraries	78
CHAPTER 5 RESULTS AND DISCUSSIONS		79
5.1	Features, scaling, and dimensionality reduction assessments	79
5.1.1	Individual features analysis	79
5.1.2	Combination of image-based features, scaling, and dimensionality re- duction	81
5.1.3	All features combined	83
5.2	Models fine-tuning	84
5.3	Simple informative comparison: Image-based features versus CNNs	86
5.4	Approach assessment	87
5.5	Comparison with literature	89
5.6	Outlier retrievals detection	90
CHAPTER 6 CONCLUSION		92
6.1	Limitations	92
6.2	Future work	92
REFERENCES		94

LIST OF TABLES

Table 2.1	Example of confusion matrix	11
Table 4.1	Statistics on the numbers of rows, columns, and pixels in a retrieval before and after applying cropping	55
Table 4.2	Statistics of plumes CH ₄ Emission Levels (in kg/h), rounded to the nearest Integer	58
Table 4.3	Statistics of retrievals CH ₄ Emission Levels (in kg/h), rounded to the nearest Integer	59
Table 4.4	Average Pearson Correlation Coefficients (r) for different channels (X,Y)	65
Table 4.5	Feature Categories and vector sizes	67
Table 4.6	Hyperparameter search spaces and best configurations for ADA, GB, HGB, RF, Bagging of DT, LR and NN	74
Table 5.1	Comparison of LR classification performances for each feature with and without scaling for class 1. F-ID, Acc, Prec.1, Rec.1 F1.1, DR.1. DR.2 DR.3, refers respectively to Feature category ID, accuracy, precision for class 1, recall for class 1, f1-score for class 1, and detection rate of retrievals in categories 1,2,3.	80
Table 5.2	Comparison of ADA classification performances for each feature without feature scaling for class 1	80
Table 5.3	Comparison of LR and ADA classification performances, with and without scaling, with and without PCA and SVD for all image-based features except features 6	82
Table 5.4	Comparison of LR and ADA classification performances for image-based features only, all features with and without satellite ID	83
Table 5.5	Comparison of LR and ADA detection rates performances for image-based features only and all features	83
Table 5.6	Comparison of the fine-tuned models classification performances . . .	84
Table 5.7	Comparison of the fine-tuned models detection rates	85
Table 5.8	Comparison of the classification performances and detection rates of ResNet architectures and HGB on image-based features	86
Table 5.9	Comparison of the fine-tuned models classification performances averaged over 5 splits	88
Table 5.10	Comparison of the fine-tuned models detection rates averaged over 5 splits	88

LIST OF FIGURES

Figure 2.1	Example of neuron for P features	16
Figure 2.2	Example of a Fully connected neural network architecture	19
Figure 2.3	Example of a simple CNN architecture	24
Figure 2.4	Example of a Decision Tree internal structure	25
Figure 4.1	<i>CH₄-SER</i> retrievals from four categories, from top-to-bottom : (1) No Plume and Low <i>CH₄-SER</i> Level, (2) No Plume and High <i>CH₄-SER</i> Level, (3) Hard-to-Detect Plumes, and (4) Easy-to-Detect Plumes. Plumes are framed. Color scheme is applied, where dark blue and dark red represent respectively very low and very high CH ₄ concentrations.	52
Figure 4.2	Visual comparison of channels. Each row corresponds to a retrieval and each column to a channel, from left to right: <i>CH₄-SER</i> , <i>CH₄</i> , <i>ALBEDO</i> and <i>CH₄-SER-CI</i> . For, <i>CH₄-SER-CI</i> , each of the five defined intensities is associated with a specific color, ranging from lowest to highest intensity: green, yellow, orange, red, and dark red. Light value of <i>ALBEDO</i> indicates high light reflectance, while dark value indicates low reflectance.	53
Figure 4.3	Cropping effects for different shapes of retrievals. Both rows represent the same retrieval before and after cropping. The dimensions, in pixels, before and after cropping are indicated from left to right: (895x842)-(478x356); (1040x838)-(603x448); (1402x1553)-(657x879); (1522x1165)-(1059x496); and (2253x1398)-(1609x846).	55
Figure 4.4	Number of retrievals of each satellite	57
Figure 4.5	Number of retrievals with at least one plume for each satellite	57
Figure 4.6	<i>CH₄-SER</i> retrievals ordered according to their emission levels. Each row illustrates images from each quartile of emission levels, arranged in increasing order of emission level from left to right.	60
Figure 4.7	<i>CH₄-SER</i> retrievals from different geographical clusters. Each row presents a cluster and retrievals that belong to it. The first row presents a cluster without plumes, and the other two present clusters with a lot of plumes.	62
Figure 4.8	Frequencies of pixel values inside plumes for the <i>CH₄-SER</i> channel .	63
Figure 4.9	Frequencies of pixel values outside plumes for the <i>CH₄-SER</i> channel .	64
Figure 4.10	Prediction pipeline	77

LIST OF SYMBOLS AND ACRONYMS

ADA	Adaptive boosting
ASM	Attribute selection measure
BOVW	Bag of visual words
CH ₄	Methane
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DT	Decision Tree
GD	Gradient Descent
GB	Gradient Boosting
HGB	Histogram based Gradient Boosting
HOG	Histogram of Oriented Gradients
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
LR	Logistic Regression
ML	Machine Learning
MLM	Machine Learning Model
NN	Neural Network
PCA	Principal Component Analysis
RELU	Rectified Linear Unit
RF	Random Forest
SIFT	Scale Invariant Feature Transform
SVD	Singular Value Decomposition
SS	Standard Scaler
SVM	Support Vector Machines
VC	Voting Classifier

CHAPTER 1 INTRODUCTION

In this chapter, we first present the importance of methane emissions monitoring and detection. Subsequently, we describe our motivations and the problem encountered by our industrial partner. We then present our objectives and conclude with the contributions of the work and an outline of the thesis.

1.1 Importance of methane emissions monitoring and detection

The 21st century poses an unprecedented challenge to humanity: combating and reducing the negative effects of climate change caused by greenhouse gases such as carbon dioxide and Methane (CH_4) [1].

CH_4 is a greenhouse gas that has a large impact on global warming and has a stronger potential than carbon dioxide [2], its emissions harm public health and reduce the production of agricultural and forestry ecosystems [3]. CH_4 emissions and concentrations continue to rise, making it the second most emitted human-induced greenhouse gas after carbon dioxide [2] and its atmospheric concentration is 2.5 times higher than in 1750 [4]. This led to coordinated efforts to reduce its emissions, such as the Paris Agreement, which was adopted by 196 countries in 2015 where reducing CH_4 and greenhouse gas emissions by 43% before 2030 is an important requirement to limit global warming impacts [5].

The increase in CH_4 is mainly caused by human activities related to fossil fuel usage, agriculture, and waste sectors, as well as by climate changes affecting its natural emission [2]. A significant part of CH_4 emissions comes from the oil and gas industry and often from undetected emissions due to abnormal operating conditions [6]. These undetected leaks contribute to a large portion of emissions and are not only a threat to the planet but also have an economic impact on the energy sector as they result in the loss of valuable resources.

They can be difficult to detect or quantify because a wide variety of CH_4 sources overlap geographically [2] and inventory methods miss emissions that occur during abnormal operating conditions [6]. For example, in 2015, CH_4 emissions from the supply chain industry were almost 60% higher than the estimation of the United States Environmental Protection Agency [6]. Accurate detection of CH_4 emissions can help operators identify and repair leaks as early as possible. This will help reduce the environmental impact because methane is removed from the atmosphere faster than carbon dioxide, which can significantly reduce the short-term environmental effect [6] and reduce the economic losses caused by leaks.

Whether leaks or regular emissions, it is important to detect and monitor them. This allows for a better estimate of CH₄ levels in the atmosphere, which helps meet regulations and sustainability commitments such as the Paris Agreement [5]. For all these reasons, CH₄ monitoring and detection has become an important area of research.

1.2 Methane emissions monitoring

1.2.1 Limits of traditional quantification methods

Traditional CH₄ quantification methods can be divided into two categories: bottom-up and top-down approaches [6]. The first produces emission estimates of geographical regions by combining several measurements made at emission sources directly from individual facilities, operations, or equipment. The second quantifies methane concentrations in the environment using aircraft-based mass balance, atmospheric models, or downwind measurement [6, 7].

Results obtained using top-down approaches often produce higher estimates than those obtained using bottom-up ones [8, 9, 10]. This is explained by the fact that bottom approaches do not consider leaks or unknown emissions, the source must be actively monitored [6]. They are also expensive, making it difficult to perform in-depth local measurements on a large scale [8] and are complex to implement in inaccessible or dangerous locations.

Top-down approaches take directly into account the emission levels in the atmosphere, making them generally more reliable. However, they are also expensive and lack precision, it is not possible to precisely identify the type of source and its location [7, 9].

1.2.2 Role of remote imagery

Airborne and satellite imagery plays an important role in monitoring greenhouse gases emissions. In 2009, Japan launched the GreenHouse gases observing satellite (GOSAT) [11], it provides important data for scientists, such as precise measurements of atmospheric methane concentration, and has been used for example in 2015 to estimate global and North American CH₄ emissions [12].

In 2017, the European Space Agency’s Sentinel-5P satellite was launched with the Tropomi instrument on board, and its effectiveness for detecting and monitoring emissions on a global scale was demonstrated by many works [13, 14, 15, 16].

GHGSAT, a Canadian startup that monitors, detects, and quantifies methane emission sources worldwide for industrial facilities, pipelines, and other areas [17, 15] launched its first satellite with hyperspectral sensors, Claire, in 2016, and has since expanded its con-

stellation to nine satellites [18]. In 2022, its constellation monitored over 500,000 potential emitting sites over an area of 2 million km² of land, from which 179 tons of carbon dioxide equivalent of CH₄ emissions were detected [18].

In 2019, Italy launched the PRecursore IperSpettrale della Missione Applicativa (PRISMA) satellite to monitor the Earth’s surface properties with hyperspectral imagery. While PRISMA’s primary mission is not specifically CH₄ monitoring, the retrieved hyperspectral data can be exploited to detect and quantify CH₄ emission sources [19].

In addition to these space-based technologies, the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) [20] an airborne hyperspectral high-resolution spectrometer, is operated by the National Aeronautics and Space Administration (NASA) to collect data on Earth’s surface and can be used to detect and quantify CH₄ emissions [21, 22, 23].

The data collected by this type of technology are used to identify emission plumes on the imagery and to quantify them using methods such as the source pixel method, cross-sectional flux method, or integrated mass enhancement [24, 23, 25, 15].

Remote sensing imagery allows the detection, location, and quantification of methane emissions from different sources and locations on a large scale. In addition, it provides a continuous stream of data that can help track the emissions over time and study their evolution daily, permitting continuous monitoring.

1.3 Collaboration and motivation

This research was conducted in collaboration with GHGSAT, an industrial partner that operates satellites equipped with hyperspectral sensors specifically designed for monitoring greenhouse gases emissions from industries and other sources worldwide [17]. The GHGSAT constellation consists of nine satellites at altitudes around 500 km: GHGSAT-D (technology demonstrator satellite) launched in 2016 and a commercial fleet (GHGSAT-C1 - GHGSAT-C8) launched between 2020 and 2023 [18].

The imagery is retrieved using hyperspectral imaging sensors collecting spectral data from the narrow band of the short-wave infrared region. These sensors have a very high spectral and spatial resolution of 0.3 nm and 25 m and capture data in several hundred thousand pixels for an average field of view or scene size of 12×12 km². Hyperspectral data are processed to produce **CH₄ enhancement images**, where the value of each pixel represents the CH₄ concentration level in a vertical column.

With very precise sensors, a high revisit frequency, and global coverage, GHGSAT satellites made it possible in 2022 to observe 500,000 potential emitting sites over an area of 2 million

km² of land, from which 179 tons of carbon dioxide equivalent of CH₄ emissions were detected [18].

GHGSAT derives the emission rate of a CH₄ plume from its total mass enhancement, length, and the wind speed [26]. However, an operator must first find manually the plumes on the retrievals before quantifying them, and because of the large volume of retrievals produced daily and the scale of the monitoring (e.g. worldwide), it results in a significant expected workload.

1.4 Scope

The work is grounded in a complex project involving multiple stakeholders. The ultimate goal of the project is to develop a complete and robust pipeline for detecting plumes. This pipeline consists of a plume detector, which segments and localizes the plumes, and a classifier that categorizes the retrievals based on whether they contain at least one plume or not.

INO Canada is tasked with developing CNN architectures aimed at detecting and segmenting plumes directly on retrievals. The team at Polytechnique Montréal is responsible for developing the retrieval classifier. Firstly, this classifier adds an additional prediction layer to the models developed by INO. Secondly, it allows the use of the *Likelihood of Computational Profiles* [27] to identify erroneous or outlier retrievals.

The *Likelihood of Computational Profiles* defines the *computational profile* of an input as a vector of neuron activation levels. It can be extracted during a neural network training phase, which allows the establishment of the distributions of the computational profiles associated with each class in the dataset. These distributions can be used during model inference to measure the likelihood of out-of-distribution and in-distribution inputs [27, 28].

In this context, two requirements are set :

1. The solution must integrate at least a deep learning method to allow the extraction and study of the *computational profiles*
2. The activation levels should describe the entire retrieval and cannot be extracted from individual regions of interest. Therefore, solutions that consider image subregions as inputs or those that classify subregions of the retrieval are excluded.

1.5 Research objective

The research objective is to develop a machine learning model capable of **performing the binary classification of methane image retrievals**: whether the retrieval contains at least one plume or not. This initial classification stage serves as a foundation, directing subsequent works such as the classification and segmentation of individual plumes.

It's crucial to note that the effectiveness of utilizing the *Likelihood of Computational Profiles* [27] to identify erroneous or outlier retrievals relies heavily on the model's performance. This approach allows the identification of flawed observations caused by factors like adverse weather affecting retrieval quality or errors in acquisition sensors.

This main objective can be divided into 4 sub-objectives :

1. Study and analysis of the GHGSAT dataset
2. Design a machine learning method based on the knowledge acquired from (1)
3. Implement and improve the method
4. Evaluate the method

1.6 Contributions

The present work contributes to the fields of machine learning, remote image recognition, and greenhouse gases emissions monitoring by proposing an approach based on the design and extraction of hand-crafted features combined with machine learning models to automatically distinguish methane retrievals that contain plumes from those that do not.

The main contributions can be summarized as follows:

- Analysis of the GHGSAT dataset under different variables, including visual aspect, pixel intensities, and retrievals geolocation.
- Design, implementation, and assessment of two categories of handcrafted features: image-based and geospatial.
- Comparison of image-based features with convolutional neural network architectures.
- Design, development, and assessment of an approach that permits the binary classification of retrievals of varying sizes using handcrafted features and that permits the use of the *Likelihood of Computational Profiles* [27] to identify erroneous or outlier retrievals.

- Comparison of the results with those of other approaches proposed in related works.

The results show that the proposed approach is viable for this problem and that emission rates are an important factor in CH₄ plume detection.

1.7 Thesis overview

The rest of the work is organized as follows :

- Chapter 2 reviews the theory necessary to understand the work and the proposed solution.
- Chapter 3 reviews the literature on image classification, smoke detection, and related works on CH₄ plume detection.
- Chapter 4 presents the proposed approach, analysis of the dataset, and assessment methods.
- Chapter 5 presents and analyzes the results.
- Chapter 6 summarizes the study and presents the limitations of the approach and future works.

CHAPTER 2 BACKGROUND AND THEORY

This chapter explains the theory necessary to understand this work and the proposed solution to the classification problem that needs to be resolved. It is mainly based on the book *Pattern Recognition and Machine Learning* [29].

Fundamental concepts are covered, ranging from machine learning basics to data preprocessing, classification tasks, parametric models, neural networks, convolutional neural networks, and model selection.

2.1 Machine learning basics

Machine learning (ML) is a field of artificial intelligence that aims to develop models capable of allowing computers to automatically learn to perform tasks from experience or data, without the need to be programmed by rules.

2.1.1 Formal definitions of a machine learning problem

A ML problem consists of resolving a task $\mathcal{T}$, using a Machine learning model (MLM) $\mathcal{M}$ and a dataset $\mathcal{D}$.

The dataset $\mathcal{D}$ is defined by the number of observations N and the number of features P it contains. Each observation d_i of $\mathcal{D}$ comprises a feature vector x_i of length P and if it is a supervised problem, a target vector y_i . All observation feature vectors can be defined as a feature matrix $\mathbf{X}$ of dimension $N \times P$ defined as follows:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}$$

Each feature vector x_i can be seen as an individual data point, described by different features, in a multidimensional space where each axis represents a different feature, such as space is called *feature space*.

The goal is to use the model $\mathcal{M}$ to map the feature matrix to the desired output $\hat{\mathbf{Y}}$, such as

$$\hat{\mathbf{Y}} = \mathcal{M}(\mathbf{X})$$

2.1.2 Supervised and unsupervised learning

Supervised learning

In supervised learning, an observation d_i in the dataset is composed of a target vector y_i of dimension S in addition to the feature vector x_i , such as :

$$\mathcal{D} = \{(x_i, y_i) \mid i = 1, 2, \dots, N\} \quad (2.1)$$

.

All observation target vectors can be defined as a target matrix of dimension $N \times S$, which is defined as follows:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad (2.2)$$

Usually, $S = 1$ and y_i are considered simple values and not vectors; they can be continuous values or categorical variables (such as a class or a label). In this case, $\mathbf{Y}$ comprises target variables and has dimension $N \times 1$.

The goal is to find the best approximation $\hat{Y}$ of the relation between $\mathbf{X}$ and, Y such as :

$$\begin{bmatrix} \hat{y}_1 = \hat{Y}(x_1) \\ \hat{y}_2 = \hat{Y}(x_2) \\ \vdots \\ \hat{y}_N = \hat{Y}(x_N) \end{bmatrix} = \hat{Y}(\mathbf{X}) \approx Y \quad (2.3)$$

$\hat{y}_i$ is defined as the model prediction for observation i .

The work focuses on supervised learning, except for the use of a clustering algorithm (subsection 4.2.7).

Unsupervised learning

$$\mathcal{D} = \{(x_i) \mid i = 1, 2, \dots, N\} \quad (2.4)$$

The goal is to use $\mathbf{X}$ to find relationships and patterns in the data $\mathcal{D}$, without using a target variable. It is generally used to reduce the dimensionality of the dataset (dimensionality

reduction) and to cluster (clustering) similar observations together on the basis of their positions in the feature space.

2.1.3 Machine learning model

Model

A MLM is a computational algorithm or mathematical representation that uses data or experiences to learn existing patterns and exploits these learnings to make predictions on unseen data. There are several types of supervised learning models, which are detailed in the remainder of this chapter.

Lifecycle of a model

The lifecycle of MLM is composed of two stages. First, is the training stage, during which the model uses $\mathcal{D}$, to learn $\hat{Y}$. The model utilizes both the feature matrix $\mathbf{X}$ and the target matrix $\mathbf{Y}$ to learn $\hat{Y}$. Once this stage is completed, the model is trained and can be employed to make predictions.

The second stage is the inference or prediction stage, in which the trained model is used to make predictions. During this step, the model only has access to the features of the observations.

2.1.4 Generalization and cross-validation

A MLM is useful only if it can provide good predictions for observations that it never encountered during the training phase. This property is called *generalization*.

Overfitting and underfitting

If the model performs very well on the training data but makes bad predictions, it is called *overfitting*. This occurs when the model perfectly learns (by heart) the relationships between the features and the target variable in the training set and goes so far as to learn the noises present in the data. This results in poor performance on unseen observations and a failure to respect the generalization property.

If the data used during the training phase presents relationships between the features and target variables, but the model cannot obtain low errors on the training set, the chosen model is not expressive/complex enough and is incapable of learning the relationships between the features and target variables. This is called *underfitting*.

Overfitting and underfitting phenomena are always related to the size of the dataset, its complexity, and the chosen model.

Cross-validation

To guarantee the generalization property, a common practice is to split the dataset into two sets. The first set is the training set and is used only during training. The second set is the test set and is used to measure the performance of the model on observations that it did not encounter during its training phase. This process is called *cross-validation*.

The test set should be processed the same as the train set (same preprocessing, same parameters ...) and be of significant size, typically between 10 and 30% of the dataset. It is also necessary to ensure that the test set represents the entire distribution of the dataset and not just a subset.

A popular cross-validation technique is *k-fold cross-validation*, which involves dividing the dataset into k subsets of equal sizes. The model is trained on $k - 1$ subsets and tested on the remaining subset. This process is repeated k times, each time the model is tested on a different subset and trained on the remaining subsets. This provides more reliable testing of the model because it is trained and tested on all data distributions in the dataset.

2.2 Classification task

2.2.1 Definition

If the desired output for a supervised problem is a class label (categorical value), the model is facing a classification task. For K classes, the goal is to assign a class label k to an observation i based on its feature vector x_i . In this type of problem, $y, \hat{y}, k \in \{1, 2, \dots, K\}$.

If there are only two classes, ($K = 2$), it is a binary classification problem, whereas if $K > 2$, it is a multi-class classification problem. An example of a classification problem is identifying the animal depicted in an image.

In a binary classification problem with classes 0 and 1, there are four cases of predictions.

The first is true positive (TP), the model correctly predicts class 1. Similarly, true negative (TN) refers to the correct prediction of class 0 by the model.

The other two cases are false positive (FP) when the model predicts class 1 incorrectly and similarly false negative (FN) when the model predicts class 0.

Table 2.1 shows all 4 cases of predictions in the binary classification problem. This type of

Predicted Class / Actual class	1	0
1	True Positive (TP)	False Positive (FP)
0	False Negative (FN)	True Negative (TN)

Table 2.1 Example of confusion matrix

matrix is called a confusion matrix and visually presents the number of occurrences of the 4 scenarios.

2.2.2 Classification metric scores

In ML, a metric score is a measure of the performance of the model. It indicates how well it performs or not and permits comparison of different models. Because of the existence of 4 prediction scenarios, there are several metrics to measure the performance of a binary classification model.

Accuracy

Accuracy is the ratio between the number of correct predictions and the total number of predictions. This metric allows knowing the proportion of correct predictions. Its value is obtained using the following formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.5)$$

Precision

Precision is the ratio between the number of correct predictions and the total number of predictions for a given class. For class 1, its value is obtained using the following formula:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2.6)$$

This measure is important when the tolerance to false alarms (FP) is low.

Recall

Recall measures the proportion of correct predictions over the total number of instances of a given class. For class 1, its value is obtained using the following formula:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2.7)$$

This measure is important when it is important to find all instances of a class and can be referred to as the detection rate.

F1 score

Finally, the f1 score considers both precision and recall and is given by the following formula :

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2.8)$$

2.2.3 Image classification task

Image classification is the process of assigning a predefined label or category to an entire image on the basis of its content, each label represents a concept that can define the content (scene, object, set of objects). In this task, each input is an image or, more formally, a three-dimensional matrix such as $x_i \in \{\mathbb{R}^{r_i \times c_i \times C} \mid i = 1, 2, \dots, N\}$, where r_i and c_i represent the number of rows and columns, respectively, and C is the number of channels.

An image x_i can be defined as

$$x_i = \begin{bmatrix} p_1^1 & p_2^1 & \dots & p_{c_i}^1 \\ p_1^2 & p_2^2 & \dots & p_{c_i}^2 \\ \vdots & \vdots & \ddots & \vdots \\ p_1^{r_i} & p_2^{r_i} & \dots & p_{c_i}^{r_i} \end{bmatrix} \quad (2.9)$$

where each entry $p_c^r \in \{\mathbb{R}^C \mid c = 1, 2, \dots, c_i, r = 1, 2, \dots, r_i\}$ is a vector of size C , providing intensity values for each of the C channels at the pixel position (r, c) .

A traditional RGB color image contains 3 channels ($C = 3$), each representing the levels of red, green, and blue, while a black and white image consists of a single channel ($C = 1$), which presents the intensity of the pixels.

2.3 Data preprocessing

Data preprocessing consists of cleaning noises and transforming raw data to make it consumable by the model. This includes handling missing values, encoding categorical variables, and scaling features.

To ensure the generalization property, preprocessing parameters should be defined from training data only and then used to preprocess both sets.

2.3.1 Feature scaling

Feature scaling is the transformation of the numerical features of the dataset so that they are on a similar scale. This is particularly important for models that are sensitive to data magnitude, such as parametric models (see section 2.4), because higher magnitude features will have a greater impact than lower magnitude ones during Gradient descent (GD) and hence appear more important to the model. Two common feature scaling techniques are often used, Standard Scaler (SS) and Min-Max-Scaler.

Standard Scaler

SS scale values of a feature to have a mean of 0 and standard deviation of 1, it assumes a Gaussian distribution of the data. The scaled vector of the feature p by SS, $Z(x^p)$, is given by the following equation, where μ^p and σ^p are respectively the mean and standard deviation of the feature p in the dataset:

$$Z(x^p) = \frac{x^p - \mu^p}{\sigma^p} \quad (2.10)$$

Because of the Gaussian distribution assumption, the distributions of the scaled features may be affected.

Min-Max-Scaler

The min-max scaler scales feature values on the basis of its minimum and maximum values in the dataset. All values are scaled in the range $[0, 1]$. The scaled vector of the feature p by Min-Max-Scaler, denoted as $V(x^p)$, is given by the following equation, where $x_{\min}^p$ and $x_{\max}^p$ are, respectively, the minimum and maximum values of the feature p in the dataset:

$$V(x^p) = \frac{x^p - x_{\min}^p}{x_{\max}^p - x_{\min}^p} \quad (2.11)$$

2.3.2 Feature encoding

Feature encoding is the process of transforming a categorical feature into a numerical format that the model can exploit. There are two common methods for feature encoding.

One-hot encoding is used to process categorical non-ordinal features. For a single feature, it consists of representing each category by a bit, this results in a vector of size C for C categories. To encode a specific category, the associated bit is set to 1 and the others to 0.

Label encoding is used for ordinal categorical features when there is a natural order between the categories. An integer is assigned for each category.

2.3.3 Dimensionality reduction

The dimension of a dataset refers to the number of features P it contains. *Curse of dimensionality* is a phenomenon that describes the issues occurring when the dimensionality of the data is very high. This leads to high computational complexity, resulting in more training time, more memory space required, difficulties in data visualization, overfitting, and even poor model performance.

Feature selection

Feature selection is a technique consisting of identifying and keeping only the subset of the most important features, those that present the greatest amount of information, therefore removing less relevant or redundant features and hence reducing the dimensionality of the data.

Transformation to lower dimension

Another way to reduce the dimensionality of the data is to use techniques that involve transforming the dataset into a space of lower dimensionality while retaining as much information as possible.

A common method for this approach is Principal component analysis (PCA). For P features and N observation, this is done by finding the $K = P$ principal components Z of the original features $\mathbf{X}$, which are linear combinations of the features that capture the most variations in the data. Such that the first principal component z_1 explains the most variance, and the second z_2 explains the second most variance. The principal component matrix $\mathbf{Z}$ is given by the following equation, where $\mathbf{Z}$ is the principal component matrix of size $(N \times K)$ and $\mathbf{A}$ is the loading matrix of size $(K \times P)$:

$$\mathbf{Z} = \mathbf{X}\mathbf{A}^T \quad (2.12)$$

For a sparse feature matrix containing a lot of zeros, PCA may perform poorly due to its linearity assumption. Singular value decomposition (SVD), another dimensionality reduction technique, can present better results in those cases.

2.3.4 Image noise reduction

Image noises are defined as unwanted and random phenomena that distort the information presented in an image and its quality. They may have several origins: sensor limitations, sensor failure, or electronic interference ... Image noise reduction is the process of reducing these noises to enhance image content.

Gaussian blur or Gaussian smoothing, is a noise reduction technique that works by convolving the input data with a Gaussian Kernel, corresponding to a matrix of weights. For each pixel to transform, the center of the kernel is placed on the pixel, and its value is replaced by the convolution of the kernel over the neighborhood of the pixel. The weights in the kernel give more importance to the nearby pixels. A smoothing parameter allows for adjustment of the blurring process, when it has a low value, only the closest pixels are taken into account, resulting in a sharper blur.

2.4 Parametric Models

Parametric models have a fixed number of parameters that define them. They are learned during the training stage, to fit the data as best as possible. The general equation of a parametric model can be stated as follows:

$$\mathbf{Y} = \hat{Y}(\mathbf{X}; \boldsymbol{\theta}) + \epsilon \quad (2.13)$$

Where $\hat{Y}$ is the model parameterized by $\boldsymbol{\theta}$, $\mathbf{Y}$ is the target matrix, and ϵ represents the error of the model. The goal of the training is to find values of $\boldsymbol{\theta}$ that minimize a cost function, which indirectly reduces ϵ .

2.4.1 Linear Model

In a linear model, a prediction is obtained by performing a weighted sum of the feature values, and by passing it through an activation function if necessary to introduce non-linearity into

the output or/and modify the output value in order to meet certain conditions (e.g. output values must be greater than 0).

The following equation defines the output of a liner model for observation i , where $\hat{y}_i$ is the output, σ is the activation function, $\mathbf{w}_p$ are the weights, b is the bias term, and $\mathbf{x}_i^P$ are the features values of observation i :

$$\hat{y}_i = \sigma \left(\left(\sum_{p=1}^P \mathbf{w}_p \cdot \mathbf{x}_i^P \right) + b \right) \quad (2.14)$$

Figure 2.1 illustrates this operation, the inputs represent the values of the features for the observation. From now on, this structure will be referred to as a neuron.

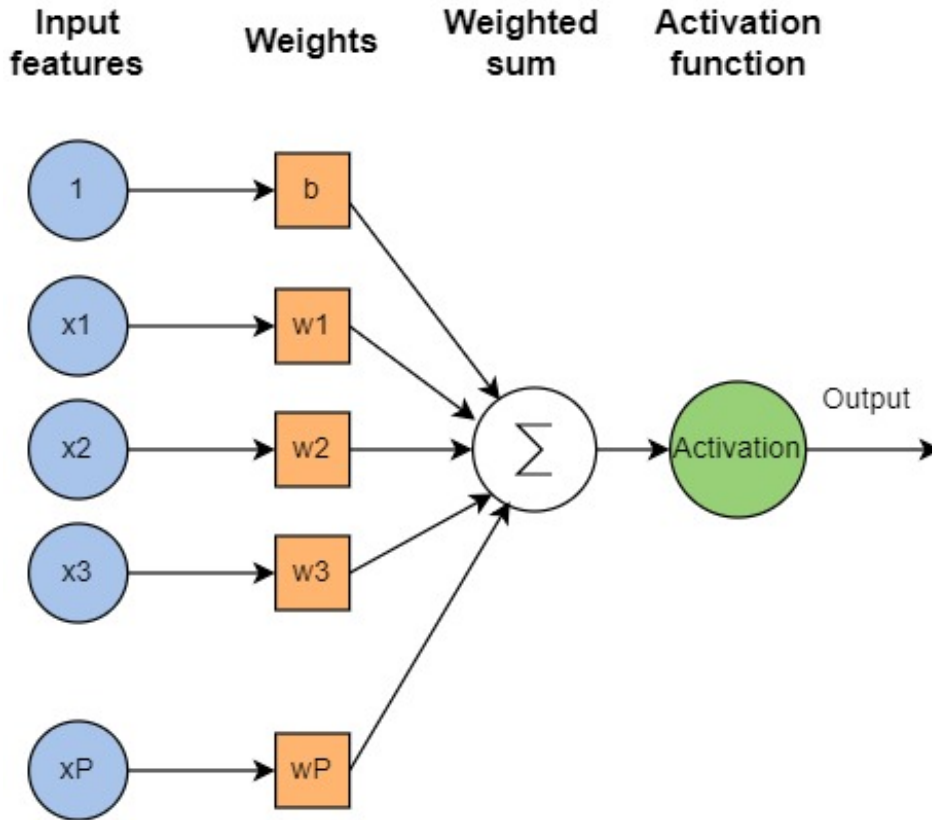


Figure 2.1 Example of neuron for P features

2.4.2 Logistic regression

Logistic regression (LR) is a type of linear model applicable to classification problems. This model uses the structure of the neuron but due to the nature of a classification problem, it

must produce a probability value between 0 and 1.

Therefore, the activation function used for a binary classification problem is a sigmoid function which squeezes all values between 0 and 1. A softmax function that returns a vector, with each component representing the probability that the current observation belongs to a class, is used for multi-class classification tasks.

2.4.3 Loss and cost function

A parametric model must learn the optimal parameters θ to make correct predictions. To do that, it is necessary to define a loss function, $L(y, \hat{y})$ which makes possible to quantify the errors of the model during the learning phase.

Binary cross-entropy is often used as a loss function for classification tasks, it is given by the following formula, where y is the true binary label (0 or 1) and $\hat{y}$ is the predicted probability of belonging to class 1 :

$$L(y, \hat{y}) = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \quad (2.15)$$

Likewise, a cost function is used to calculate the error for all N samples in the training set. The following equation shows the cost function associated with the binary cross-entropy loss, where N is the number of samples:

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N [y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})] \quad (2.16)$$

The goal is to find the parameters θ that minimize the cost function. This is often done using the GD optimization algorithm that is used to find the minimum of a function.

2.4.4 Gradient Descent

GD is an algorithm used to find the minimum of a given function. It is based on the concept of the gradient of a function, which is a vector of partial derivatives that quantify the rate of change of the function with respect to each parameter. The gradient indicates the direction and the magnitude of the steepest increase in the function.

For P parameters, the gradient of the function J with respect to the vector of parameter θ is given by the following equation, where $\frac{\partial J}{\partial \theta_i}$ represents the partial derivative of J with respect to the parameter θ_i : $\nabla J(\theta) = \left[\frac{\partial J}{\partial \theta_1}, \frac{\partial J}{\partial \theta_2}, \dots, \frac{\partial J}{\partial \theta_P} \right]$.

Algorithm 1 details the gradient descent algorithm. First, the model parameters are initial-

ized, afterwards, the algorithm iteratively computes the cost function $J(\boldsymbol{\theta})$ and its gradient. Then updates the parameters $\boldsymbol{\theta}$ in the opposite direction of the steepest change, this phase is called parameters update. It stops when it reaches convergence or meets a stopping criterion.

Algorithm 1 Gradient Descent

```

1: procedure GRADIENTDESCENT( $\alpha, J, T_{\max}, \epsilon$ )
2:   Initialize  $\boldsymbol{\theta}$  randomly or with predefined values.
3:   Initialize a variable  $t$  for the current iteration to 0.
4:   while  $t < T_{\max}$  or  $J(\boldsymbol{\theta})$  not converged do
5:     Compute the gradient:  $\nabla J(\boldsymbol{\theta})$ .
6:     Update the parameters:  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla J(\boldsymbol{\theta})$ .
7:     Increment  $t$  by 1.
8:     if  $\|\nabla J(\boldsymbol{\theta})\| \leq \epsilon$  then
9:       Break (convergence criterion met).
10:    end if
11:  end while
12:  Output: The optimized parameters  $\boldsymbol{\theta}$ .
13: end procedure

```

α is the learning rate, that controls the size of the steps taken during the parameter updates. A smaller learning rate leads to slower convergence but is more stable, while a larger one may lead to faster convergence but can be unstable and ϵ is the convergence criterion.

2.5 Neural Networks and deep learning

Neural network (NN) is a type of parametric MLM made up of neurons organized in layers, where neurons from consecutive layers are interconnected in such a way that the neurons of a layer take as input the output values of the neurons from the previous layer.

Figure 2.2 shows an example of a fully connected neural network architecture, the nodes represent the neurons and the edge represents the weights. In a fully connected NN, each neuron of a layer is connected to all the neurons from the previous layer.

2.5.1 Architecture

There are 3 types of layers in all neural networks: input layer, hidden layers, and output layer.

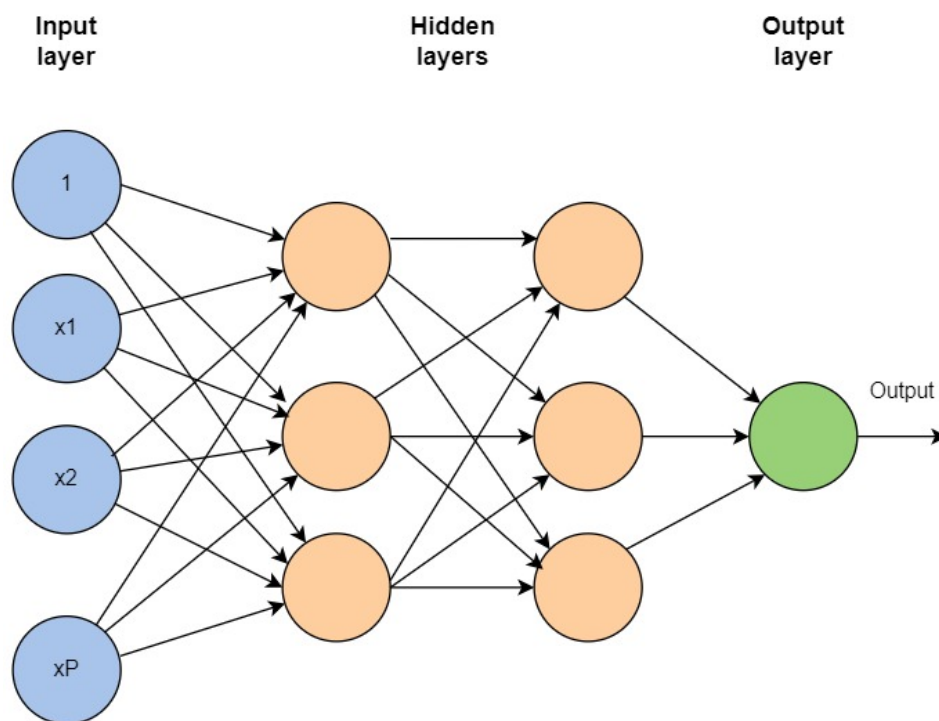


Figure 2.2 Example of a Fully connected neural network architecture

Input and output layers

The input layer is a representation of the network input data, each node in this layer is associated with a feature, they do not perform any computation and only pass the feature values to the next layer.

The output layer is connected to the last hidden layer and is responsible for making the model predictions.

Hidden layer

Hidden layers are responsible for processing the data and learning the patterns present in it. There must be at least one hidden layer in a neural network, but there can be several. The depth of a neural network refers to the number of hidden layers it has.

Deep learning

Deep learning refers to the use of a NN with at least two hidden layers. Deep NN are capable of learning hierarchical representations of the data. Indeed, the complexity of the relationships and patterns learned by a hidden layer gradually increases with its position in

the network.

A hidden layer close to the input layer learns simple relationships relative to those close to the output layer, which learn more complex and abstract patterns. The more abstract the patterns, the less sensitive they are to small variations in input values. This phenomenon is called hierarchical feature learning. This also permits features to be automatically learned from raw data with only a bit of preprocessing, this is called automatic feature learning.

2.5.2 Learning process

A NN learning process consists of the alternation of two phases implementing the GD algorithm: feedforward propagation and backpropagation. Once the model parameters defined after training, the observations to be predicted pass through feedforward propagation to obtain the model predictions.

Feedforward propagation

Feedforward is the process by which data passes through the neural network, from the input layer to the output layer. It is a sequential process, where the outputs of one layer are computed and used as inputs for the next layer.

Backpropagation

The backpropagation is the method for NN to update its parameters using the GD algorithm. Once the forward propagation is complete, the model loss is calculated. The partial derivatives of the loss function are then calculated for each parameter, allowing them to be updated, starting with the weights of the neurons in the output layer up to the weights connecting the input layer and the first hidden layer.

Other concepts

Here, several concepts related to NN training are given :

- An **epoch** refers to one pass over all the data in the training set. The number of epochs defines how many times the model encounters all observations during the training phase.
- A **batch** refers to the number of observations that pass through the model in parallel during the training phase, the model weights are updated after each batch.

- **Batch GD** is a learning method where a batch is composed of the whole training set (one epoch), it is computationally expensive.
- **Stochastic gradient descent (SGD)** is a learning method where a batch is constituted of one observation.
- **Mini batch GD** is a learning method where a batch is composed of at least two observations and is smaller than the whole training set.
- **Learning rate scheduling** is a learning method where the learning rate is updated during the training phase. There are several methods of learning rate scheduling.
- **Transfer learning** consists of initializing the model with the weights of a model already trained on a similar task, this helps speed up the training process.
- **Dropout technique** is a regularization technique consisting of randomly deactivating some neurons during each training iteration, it makes the model more robust by avoiding relying too much on a subset of neurons [30].
- **Batch normalization** is a technique to speed up and improve the stability of the learning phase. This helps alleviate the covariate shift problem, where neuron activation levels during the training phase vary too much from batch to batch. Batch normalization introduces two additional learnable parameters for each neuron (mean and variance), responsible for controlling its normalization [31].
- **Regularization:** is a practice used to prevent overfitting for parametric models. It generally consists of adding penalties associated with the parameter magnitudes to the loss function, encouraging the use of simple parameters.

2.5.3 Activation functions

The main role of activation functions is to introduce nonlinearity into the NN, which allows learning complex nonlinear relationships, it makes the network more expressive. Here, some popular activation functions are presented.

Sigmoid function

The Sigmoid function is given by the equation $\sigma(x) = \frac{1}{1+e^{-x}}$. It squashes its input into the range [0-1], so it can be interpreted as probabilities and can be used as an activation function for the output layer in a binary classification problem.

However, the range of values of the sigmoid derivative is $[0-0.25]$, so it can lead to the *vanishing gradient problem*. This phenomenon happens when the gradient is very small, during the backpropagation, especially when the network is very deep. It can slow down or even stop the weights update process.

Hyperbolic Tangent function

The Hyperbolic Tangent function is given by the equation $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. This function is very similar to the sigmoid but squashes its input into the range $[-1-1]$. Like the sigmoid, it can lead to the vanishing gradient problem.

Rectified Linear Unit (RELU) function

RELU is given by the equation $\text{ReLU}(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases}$.

It returns the input if it is positive, otherwise, it returns 0. The advantage of this function is that it is very quick to compute. However, it can lead to the problem of *dying RELU*, when certain neurons always output zero during training, never get updated, and never activate.

A solution for this problem is to use variants like Leaky RELU or Parametric RELU, which allow a small, non-zero gradient for negative inputs.

2.6 Convolutional Neural networks

A Convolutional Neural network (CNN) is a NN designed to work with images as inputs. This is achieved using two types of layers in addition to those found in a traditional NN: convolution layers and pooling layers.

2.6.1 Convolution layer

Convolution

Convolution is an operation involving sliding a filter/kernel over the input data. A filter can be seen as a matrix of learnable weights and is responsible for learning a specific pattern from the input (horizontal edges, vertical edges ...).

The convolution operation consists of sliding the filter across the entire image and computing the dot product between itself and a local region of the input image. The following equation

represents the convolution process :

$$(C * K)[i, j] = \sum_m \sum_n C[i - m, j - n] \cdot K[m, n] \quad (2.17)$$

$(C * K)[i, j]$ is the value at position (i,j) in the output of the convolution, C is the input image and K is the kernel. Each resulting value $(C * K)[i, j]$ is passed through an activation function to break linearity.

It produces a *feature map*, where each value represents the result of the convolution of the filter with a region of the image.

Specificity

Each value of the feature map can be seen as a neuron connected to a local region of the input image, this is called local connectivity. This permits to reduce the number of parameters compared to a traditional network where each neuron would be connected to each pixel.

Another concept that allows the reduction of the number of parameters is weight sharing: a filter has the same weights for each region of the image. This also allows the same pattern to be captured across the entire image, making the CNN translation invariant.

The parameters that influence the convolution process are the use or not of padding, the size of the filter or the receptive field, and the stride which indicates the step to be taken in the sliding process during the convolution.

A convolution layer can be composed of multiple filters, each applied to the input, resulting in a feature map for each filter.

2.6.2 Pooling layer

Convolution layers are usually connected to pooling layers. Pooling is a subsampling operation used to reduce the size of the feature map while preserving as much information as possible and reducing sensitivity to minor differences and overfitting.

Pooling involves sliding over windows of the feature map and keeping for each one: the average value (average pooling) or the maximum value (maximum pooling). A pooling layer will produce an output image for each feature map produced by the convolution layer that precedes it.

2.6.3 Architecture

A CNN consists of two parts. The first one is the feature extractor, which is composed of a succession of convolution layers and pooling layers. These two types of layers have the role of reducing the computational complexity and permit to learn and extract (automatic feature learning) from raw image data meaningful features such as edges, textures, and more complex patterns.

This part is followed by the classifier, which is a traditional NN that takes the flattened outputs of the feature extractor as inputs. Figure 2.3 illustrates a simple CNN architecture, with one convolution layer, one pooling layer, and two fully connected layers.

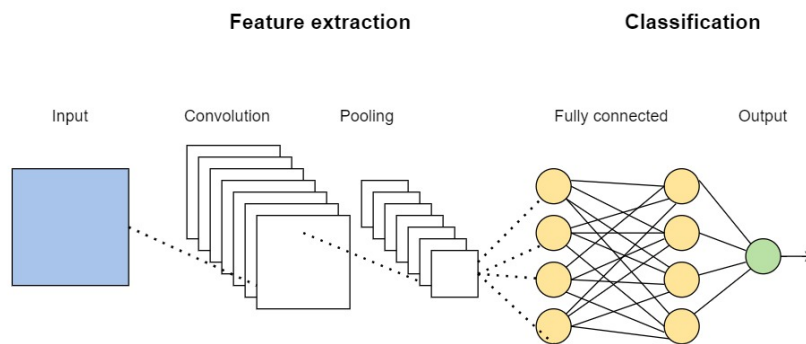


Figure 2.3 Example of a simple CNN architecture

2.7 Decision Tree

Decision Tree (DT) is a type of MLM that implements a tree structure to learn how to make predictions. In the learned tree, each internal node represents a test on the value of a feature, each branch represents the output of a test, and each terminal node represents a prediction. To make a prediction, the algorithm uses the values of the features of the observation to traverse the tree, until arriving at a leaf that contains the prediction to be given. Figure 2.4 shows the internal structure of a DT, internal and terminal nodes are respectively in yellow and blue.

A DT is constructed by recursively splitting the training data into subsets using a split rule based on the value of a feature. The best features and associated separation rules are found with an Attribute selection measure (ASM).

ASM is a criterion used to select the best features to do the split based on metrics such as Gini impurity or entropy. Those metrics measure the level of heterogeneity of the resulting

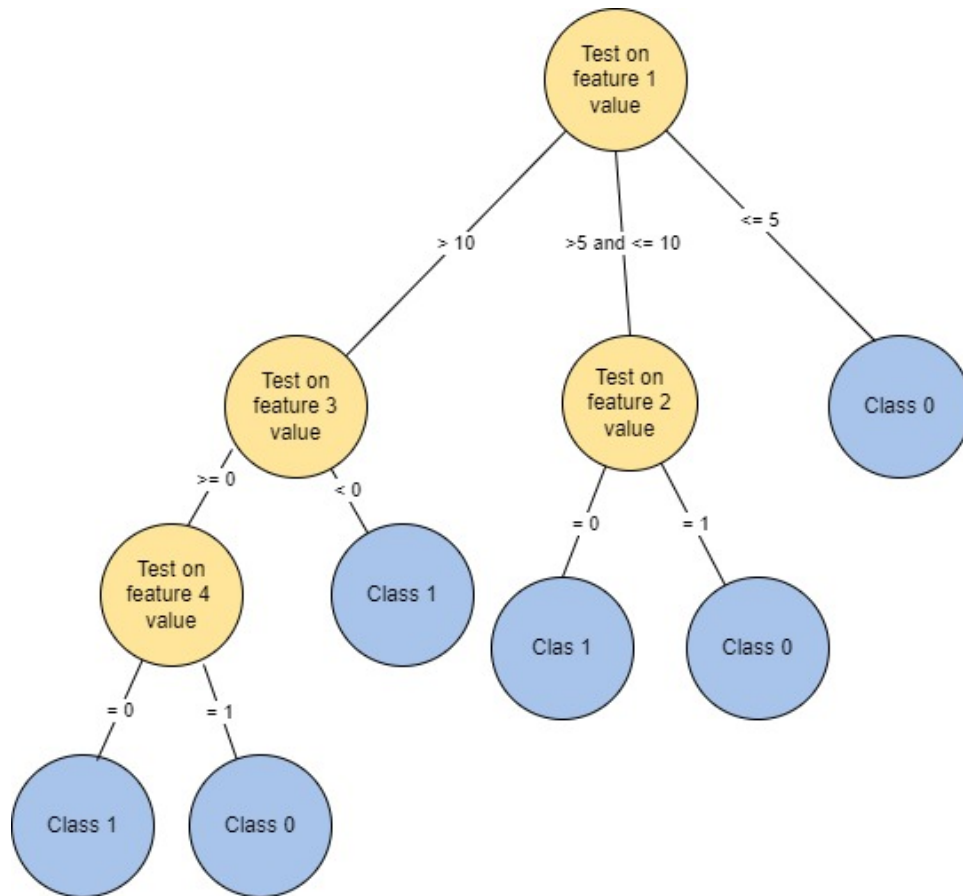


Figure 2.4 Example of a Decision Tree internal structure

subsets with respect to the target variable. The goal is to identify the feature and the split rule that will create the most homogenous subset splits. Once a subset contains only the same label, it is no longer split, the contained label then becomes the prediction for this leaf.

DTs are capable of learning complex patterns and provide explainability, in fact, once the training is completed, it is possible to extract the decision rules to visualize them and interpret the reasoning.

2.8 Ensemble methods

Ensemble methods are ML techniques that combine predictions from several models in order to increase prediction performances and reduce variance. The resulting model is called a strong learner, and the models that compose it are called weak learners or meta-learners.

There are 3 main types of ensemble methods: bagging, boosting, and stacking. In a bagging or a boosting procedure, weak learners are different instances of a simple model (LR, DT,

etc.), on the other hand, stacking exploits meta-learners which are strong models of varying natures.

2.8.1 Bagging

Bagging or bootstrap aggregation is an ensemble method technique, which trains many weak learners on different subsets of the dataset in parallel and then merges them to get a strong learner [32].

Bootstrapping

The first step in bagging is bootstrapping, which is a resampling technique aimed at creating multiple datasets of the same size as the original dataset. To do that, observations are randomly selected from the original dataset with replacement, so the resulting datasets may present the same samples multiple times.

This random phenomenon creates a different dataset, with different patterns, for the training of each weak learner. Each one of them is affected differently by the noises and the outliers in the data, and combining their predictions can make the ensemble more robust.

Aggregation

The second step of bagging is aggregation, each weak learner makes a prediction and the strong learner combines them in a voting process to give a robust and accurate prediction. Bagging is often used with DTs (bagging of DT) as weak learners because they are very expressive and can learn complex relationships, but they also have high variance, which makes them unstable and affected by noises.

Random Forest

Random Forest (RF) is a bagging algorithm in which weak learners are DTs [33]. Its particularity lies in the learning processes of the DTs, the ASM is restricted to a random subset of features for each split. This results in the fact that each DT will create a split of the data based on different features.

In classic bagging, each DT has access to all the features to perform the ASM at each split, this can be problematic because even if the training sets differ slightly from one DT to another due to bootstrap, the separation of data is likely to be done on the same features, resulting in very similar trees.

2.8.2 Boosting

Boosting is an ensemble method technique, where weak learners are trained sequentially in such a way that each weak learner corrects the errors made by the previous one.

Traditional boosting

Traditional boosting algorithms assign more weights to the training samples that were misclassified by the previous weak learner. This way, more importance can be assigned to observations that have been misclassified.

After training, each weak learner has a weight associated with his prediction based on his performance on the training set, weak learners who performed well have a greater impact on the final result which is a weighted vote.

Adaptive boosting (ADA) is a popular algorithm that formalizes the traditional boosting procedure [34].

Gradient boosting

Gradient boosting (GB) is a boosting method based on the concept of GD, with a loss function that needs to be minimized [35]. In this method, the loss function J is parametrized by the predictions as $J(\hat{\mathbf{Y}})$, and the gradient of the loss function is calculated with respect to the predictions.

The first weak learner F_0 is trained on the target variables. The gradient of the loss with respect to the predictions is computed $-\frac{\partial J(x_i)}{\partial F_0(x_i)}$ for each observation x_i . These gradients R_0 are called residuals and are given to the next learner F_1 as target variables.

This way, the predictions of the ensemble for two weak learners are given as $G_1(X) = F_0(X) + \alpha_1 * F_1(R_0)$, where $F_1(R_0)$ is the predictions of the residuals from the second learner and α_1 is the learning rate of the second weak learner. The third learner takes as target variables R_1 (the residuals of $G_1(X)$) and this process continues until convergence.

For $M + 1$ weak learners, the ensemble predictions are given as :

$$G(X) = F_0(X) + \sum_{m=1}^M \alpha_m F_m(R_{m-1}) \quad (2.18)$$

Histogram based gradient boosting (HGB) is a variant of GB algorithm permitting to efficiently process large data sets, for which the classic GB procedure can be too demanding.

This method uses the concept of a histogram, a data structure for grouping the values of a feature into bins based on their distribution. This allows continuous features to be discretized, resulting in faster calculation.

2.8.3 Stacking

Stacking is an ensemble method that, like bagging, trains several models (meta-learners) in parallel. Their predictions on the training set are used as the strong learner feature space, which learns to combine them with the target variables to achieve the best performance.

A variant of stacking is the voting model, which does not train strong learners on meta-learners predictions but uses them as votes for the final output. There is hard voting which consists of using the most frequent class predicted by the models as output and soft voting which uses the probabilities associated with class predictions across all models and averages them to get the final output.

2.9 Hyperparameters and model selection

2.9.1 Hyperparameters

Hyperparameters are the model's configuration settings, they are not learned from the training data and are set before the training phase.

Here are the basic hyperparameters for the models presented:

- The loss function
- For parametric models: the regularization type and strength
- For DT based methods: the maximum depth, number of maximum features to take into consideration for the splits and the ASM
- For ensemble methods: the number of estimators
- For GD methods: the learning rate
- For NN: architecture (number of layers and neurons by layer), the learning process (batch size, type of GD), activation functions

2.9.2 Model selection

Model selection is the process of selecting the best hyperparameters for a given model. This involves training the model with different hyperparameter configurations and testing its performance. However, hyperparameter configurations can't be tested using the test set, as this would compromise the generalization property.

Therefore, during the model selection phase, a validation set is used to assess and select the best hyperparameter configuration, and the final performance of the model is evaluated on the test set. Appropriate hyperparameter configuration can be found using different methods.

Grid search tests all possible configurations of hyperparameters existing in a defined space, it guarantees the discovery of the best configuration in this space. Random Search randomly tests configurations in a defined space, it does not guarantee the discovery of the best configuration, but is a good alternative when computing resources are limited.

CHAPTER 3 Litterature review

First, the field of computer vision and some of its applications are briefly introduced. Second, the state of the literature regarding image classification and related tasks is presented. Third, hyperspectral data is introduced because GHGSAT methane images are retrieved from hyperspectral data. Fourth, as the literature on automatic CH₄ plume detection is limited and recent, automatic smoke detection is studied. Finally, related works on methane plume detection are discussed.

3.1 Computer vision

Computer vision is a field of computer science and artificial intelligence that aims to enable machines to interpret and understand visual data such as images or videos. Generally, a task of computer vision includes :

1. Preprocessing: This involves operations such as sampling, resizing, cropping, padding, noise removal, and normalization . . .
2. Information extraction: This consists of representing and extracting the information contained in the visual data (features extraction)
3. Patterns recognition: This implies interpreting the extracted information to make logical inferences
4. Decision: This consists of using patterns recognition in order to produce an output that varies depending on the nature of the task

Computer vision has several areas of application and is increasingly emerging as a viable technology to solve various problems in different domains, such as: healthcare [36], face recognition [37], traffic monitoring [38], fire detection [39], smoke detection [40, 41, 42, 43, 44, 45], methane detection [46, 7, 22, 47, 21, 48], ground surface classification [49, 50, 51, 52], image indexing [53], digit recognition [54], object recognition and detection [55, 56, 57, 58].

3.2 Image classification and related tasks

This section aims to provide a simple, non-exhaustive overview of the dominant image classification methods. It was performed mainly (but not only) by examining two computer vision

competitions: ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [57] and PASCAL VOC [55], which are also datasets. First, a brief introduction to the two competitions is given. After that, some important traditional feature extraction methods are presented before finishing with the automatic approaches.

3.2.1 Pascal VOC and ImageNet Large scale visual recognition challenge

Pascal VOC is a publicly available dataset of annotated images [18] and an annual competition. The PASCAL VOC competitions took place every year from 2006 to 2012. It consists of five challenges: image classification, object detection, object segmentation, action classification, and person layout [55, 56] and is still considered a benchmark dataset, with 20 object categories and over 10,000 images annotated with bounding boxes.

Similarly, ILSVRC is a dataset and competition, but on a much larger scale. The ILSVRC competition was held annually from 2010 to 2017 [57, 58]. The dataset contains over a million labelled images with multiple categories. It is an important actor in the field of computer vision.

3.2.2 Traditional feature extraction approaches

Traditional image classification techniques rely on the extraction of human-defined or hand-crafted features. These features aim to capture various types of information in an image, such as shapes, textures, colours, and statistics on pixel intensity values. Most features can be extracted globally for the entire image (global descriptor) or specifically for different regions/patches or points of interest in an image [56, 57, 58] (local descriptor). They are extracted and used to train supervised machine learning classifiers [56, 58].

Intensity features

Intensity features capture information about pixel intensities and statistics in an image or regions of interest. A popular feature is the colour/intensity histogram, the colour space is quantified into bins before counting the number of pixels in each bin.

Edge Features

Edge features describe the shapes of an image or region of interest. Image gradients measure the amount of directional change (x and y axis) and indicate the speed at which pixels change from one intensity level to another. They are often used to identify edges (large gradients).

A popular feature is HOG [59], which describes image patches with histograms where the bins represent gradient orientations and gradient magnitudes are weights.

Texture features

Local binary pattern is a texture descriptor commonly used in computer vision. It describes the texture of an image by comparing the intensity value of a central pixel to its neighbors. This operator encodes each pixel based on this comparison with a binary code. The local binary code of a pixel with a square neighborhood of 8 pixels is given by:

$$LBP(x, y) = \sum_{n=0}^7 s(f_n - f_c) \cdot 2^n \quad (3.1)$$

Where (x, y) are the coordinate of the pixel to evaluate, f_c its intensity, f_n the intensity value of the n neighbour, and $s(x)$ is a step function defined as $s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases}$

Scale Invariant Feature Transform (SIFT)

SIFT is an algorithm used to automatically localize and describe scale, rotation, and illumination invariant key points in an image. A key point is defined by its position, scale, and rotation.

This first step is called key point detection and consists of finding the key points. For each identified key point, the SIFT feature descriptors consist of a local HOG, describing the sub-regions surrounding the key point. [60].

SIFT was very used in both challenges, often with Bag of visual words (BOVW) representation or fisher vectors.

BOVW

BOVW is an image representation technique inspired by the text-based bag of words [61]. First, local descriptors (it can be any feature) are extracted from different regions of interest across all images in the training set. Subsequently, these regions are clustered based on the position of their local descriptors in the feature space, usually done by *k-mean clustering* algorithm [56], such that the regions representing the same patterns are in the same cluster.

Each cluster becomes a visual word and represents a specific visual pattern, a vocabulary is formed with all the clusters. All the images are encoded using this vocabulary.

The extracted feature vector of an image consists of k features, where k is the number of words in the vocabulary. Each entry $\{i \mid 1 \leq i \leq k\}$ in this vector represents the number of regions of interest in the image belonging to the cluster i , a region of interest belongs to the cluster to which it is closest.

The generated feature vectors are then exploited by machine learning models. This method was very used in all PASCAL VOC challenges.

Fisher vector representation

Fisher vector representation is an image representation technique similar to BOVW. The main difference resides in the fact that instead of attributing each region of interest to the nearest cluster, it describes each region of interest with its proximity to each cluster. In such a way that each region of interest is associated with different clusters (instead of just one) with different degrees, depending on the proximity between its feature descriptors and that of the word/cluster [62].

This vector representation performed strongly with different MLM each year of the ILSVRC challenges.

3.2.3 Automatic feature extraction approaches

Image classification

CNN was first introduced in 1998 in the paper named *Gradient-based learning applied to document recognition*, with an architecture named *LeNet-5*, composed of 2 convolution layers, 1 max pooling layer, 2 fully connected layers, and RELU activation function. It achieved state-of-the-art performances for digit recognition at the time it was published [54].

ILSVRC 2012 was a turning point in computer vision when a large scale CNN was introduced and won the competition. From this point on, the majority of participants started to use CNNs, which showed the best results. Over the years, many CNN models won the annual competition each time with better performances.

AlexNet, a CNN architecture won the 2012 edition. It uses a deep architecture with 60 million learnable parameters, composed of 5 convolution layers, 3 fully connected layers, a RELU activation function after each convolutional layer, and max-pooling [63].

VGGNet a simple and practical CNN architecture was proposed in 2014. It consists of the use of a simple structure: a convolutional layer with small filters followed by pooling layers [64]. This structure is repeated several times and requires many learnable parameters, more

than 120 million, depending on the depth of the architecture. It ranked second for image classification and first for single object localization in 2014.

GoogLeNet or *Inception* architecture was introduced by Google researchers in 2014 [65] and won the classification and object detection tasks and ranked second in single object localization in the 2014 edition. The concept of the inception modules is introduced, which consist of multiple parallel convolutional and pooling operations of different sizes, allowing to learn features at different scales.

Instead of fully connected layers at the end of the network, global average pooling is used, which computes the average value for each feature map. This allows to reduce the number of learnable parameters and overfitting. Additionally, *auxiliary classifiers* are injected in the middle of the network in order to mitigate the vanishing gradient problem. This architecture presents about 20 million learnable parameters.

ResNet or *Residual neural network* won the 2015 image classification task [66]. It introduced the concept of *skip-connections*, which allows passing the output of a layer to the next layer but also to a deeper layer. These skip connections permit the use of very deep NNs without encountering the vanishing gradient problem that usually occurs with deep architectures.

Object localization

Recent approaches for framing objects by bounding boxes or for identifying all the pixels belonging to an object by a mask (segmentation) rely on the use of CNN.

The *Region-based Convolutional Neural Network* (R-CNN) was introduced in 2015 with the aim of detecting, classifying and framing [67] objects in images. It relies on three different processes. First, a set of regions of interest is extracted as potential bounding boxes using a region proposal algorithm. Second, high-level visual features are extracted by a pre-trained CNN. Third, these features pass through a set of trained classifiers, each of which detects a specific object, making it possible to predict whether the region of interest contains an object, and which one. Because of all these steps, this model requires a lot of computing power.

This is improved in a more recent version *faster-R-CNN* [68] where all three stages are included and optimized within the same network. A CNN backbone is used to produce a feature map, reduce computation, and extract visual features. Then, a region-proposal subnetwork is responsible for identifying regions of interest. The feature map of each region is further processed by fully connected layers and fed to a bounding box regressor and a classifier, respectively responsible for refining bounding boxes and assigning them a class.

Mask-RCNN introduced in 2017 allows segmenting the pixels belonging to each object, in addition to providing bounding boxes and classification [69]. It is based on the same architecture as faster-RCNN and includes an additional CNN subnetwork, the mask-head, responsible for segmenting each pixel in the bounding box (background or object).

3.2.4 Takeaway

By the exploration presented in this section, the following conclusions are drawn.

To achieve a classification of input images, meaningful features are extracted to describe them and learn from them. In traditional approaches, these features are handcrafted (defined by humans) and cover several types of characteristics: color histogram, texture descriptor, edge detection, and statistics extraction . . . Then, they are extracted and encoded in a meaningful manner and learned by traditional ML methods.

Most recent approaches use CNN to learn relevant features automatically from the images. ILSVRC competitions have shown that CNNs perform better than traditional methods. However, even if they do not exhibit state-of-the-art performances, traditional methods produce acceptable results and present successful applications.

3.3 Introduction to hyperspectral data classification

GHGSAT methane images are recovered from hyperspectral data, therefore, this type of data, their use, and the difficulties they present are presented.

Hyperspectral images are data that capture hundreds and thousands of spectral bands in a high spectral resolution in association to a spatial extent. They are often presented as cubes, where the rows and columns dimensions represent the spatial extent and the third dimension represents the spectral data [49, 50, 51, 52]. More simply, they can be seen as an image, where each pixel position is associated with a spatial position, and with each channel associated with a narrow band of the electromagnetic spectrum.

A common use of this type of data is land cover classification, the goal is to detect the different types of material present in different regions of the image [52]. It contains a lot of information, and its use with ML therefore provides better results than traditional methods [49, 50, 51, 52]. However, they present several constraints.

First, the high number of spectral channels and information gives rise to the curse of dimensionality. Second, there are limited available labelled data. Third, the large spatial variability of spectral signatures can make the classification harder [49, 50, 51, 52], the same material

can have several spectral signatures. Other difficulties include the nature of the surrounding environment, sensor quality, sensor orientation, illumination, the impact of the time and season of the acquisition, and differences in atmosphere [49, 52]. A small variation in these parameters can induce large variations in the hyperspectral data [52].

ML methods with hyperspectral data involve the use of traditional models such as LR, k-nearest neighbor, ensemble methods, Support vector machine (SVM), Gaussian models, coupled with dimensionality reduction techniques such as PCA or band selection method [52]. Often, only the spectral dimension is used [49, 50, 52].

In 2014, [49] introduced deep learning-based feature extraction for hyperspectral data classification with stacked auto-encoders. Two methods are proposed. The first focuses on hyperspectral information, it consists of using stacked autoencoders on the spectral dimension to extract features and uses LR for classification. The second method is spatially focused, it consists of using PCA on the spectral dimension to reduce its dimension, then employing autoencoders on the spatial dimension and the reduced spectral dimension to extract deep features. The classification is done again with LR. By combining the two methods, better than state-of-the-art results are obtained, however, it requires a lot of learning time.

In 2016, [50] proposed a similar approach. First, spectral features are extracted using *Balanced Local Discriminant Embedding* to find a low-dimensional representation of the data. Spatial features are then extracted using a CNN framework. Both spectral and spatial features are stacked and used as inputs for a LR. This method shows very good results on a popular dataset.

3.4 Automatic smoke detection

Given the lack of research on automated CH₄ plume detection and the similarities between the properties of CH₄ and smoke [46], the study of automatic smoke detection is being explored as a pathway to address CH₄ detection.

3.4.1 Traditional approaches

In 2007, [39] proposed a method to detect smoke in photos based on pixel colors and intensities. A color model based on statistical analyses of intensities is developed. Based on the observations that smoke pixels should have similar intensities in the RGB spectrum, and color saturation should be very low, a threshold on the *Hue*, *Saturation*, *Value* (HSV) color space is used to separate pixels containing smoke from those that do not.

In 2012, [40] proposed an approach to detect smoke emissions in videos. First, frames are split into blocks and those that may potentially contain smoke are identified by using background subtraction and thresholding the saturation in the HSV color model. These thresholds are experimentally determined, similarly to [39]. Second, the blocks' spatial energies are computed by calculating the wavelet spatial energy of each pixel and averaging the values. Similarly, the spatio-temporal variation of the energy in a sequence of frames is computed. A SVM classifier is then trained on these spatio-temporal features to assess the presence of smoke. Third, Histograms of gradients and optical flow descriptors are extracted to describe the motion of the smoke, and BOVW is generated from these descriptors using k-means clustering. Another SVM classifier predicts if the block contains smoke using these features. Finally, a dynamic score combination by mean value is computed to assess the presence of smoke in a block. This approach obtained an average of 93.7% true positive rate over 20 videos.

In 2012, [41] proposed a two-step feature extraction method for smoke detection in videos. First, several sets of partitions that allow a video frame to be divided into several blocks of the same size are defined. Using multiple partition sets permits block divisions of different sizes from the image, resulting in a multiscale partitioning scheme that permits the extraction of features at different scales. Subsequently, for each partition, four types of features are extracted for each block: concatenation of HOG, local binary pattern, edge amplitudes, and color intensity histograms. This way, each block of each partition is characterized by a feature vector composed of the concatenation of these features. Second, statistical measures (mean, variance, skewness) and hu moments of these features are computed for each partition, resulting in a large feature pool. Finally, ADA is trained on this feature pool (obtained with 50 partitions). Training was done on about 28,000 images, with only 559 of them containing smoke, testing was done on a dataset having a similar number of images and a similar balance of labels. Results have shown that the method has better generalization performance and is less sensitive to geometry transformation (rotation) than other traditional methods.

In 2016, [42] proposed a method based on a local ternary pattern which is an extension of the local binary descriptor. First, the directional derivatives/gradients of multiple orders are computed, and local ternary patterns are generated from them. The histograms of the different derivative orders are joined, resulting in a High-order local ternary pattern. *Locality-preserving* projection is used to reduce the dimension and thereby the computation time and noisy derivatives are removed by thresholding. Second, the average magnitude for each of the derivative orders is computed, and each pixel is encoded using its neighborhood derivative's magnitudes and the average magnitude of the derivative order in a similar way to the local binary pattern. Third, the average pixel value for the entire image is computed, and each pixel is encoded by comparing it to this value. This way, each pixel has three types of features:

high-order local ternary patterns, local magnitude patterns, and local center patterns. These features are concatenated to obtain a histogram, and a SVM is used to do the classification. It was trained on about 1400 cropped images, with 40% of them containing smoke. The approach was tested on about 24,000 images and had an average detection rate of 94%.

3.4.2 Deep learning approaches

In 2016, [43] proposed a simple CNN approach to detect smoke emissions on 64×64 RGB images. The CNN architecture is composed of four convolution layers, two max pooling layers, and two fully connected layers. Dropout regularization is implemented to reduce overfitting. The model was trained using mini-batch SGD, with a batch size of 100 on approximately 16,500 images, with 30% of them containing smoke. A classification accuracy of 96% and a detection rate of 98% were achieved.

In 2017, [44] proposed a CNN architecture called Deep normalization and convolutional neural network (DNCNN) for automatic smoke detection. The first eight layers are an alternation of convolution and normalization layers, with 3 pooling layers and the last three layers are fully connected layers that implement dropout regularization to avoid overfitting. The training is done using mini-batch SGD, the momentum trick and learning rate decay after each epoch, and the loss function is the cross-entropy. The only preprocessing done on the images is min-max normalization. The architecture has 20 M learnable parameters, which is significantly less than popular CNN architecture like AlexNet or VGG.

First, the importance of normalization layers was proved by training a similar model without them and comparing the results. Second, the model is trained on three datasets, the original one and two others that were obtained by performing data augmentation on the training set (horizontal flip, vertical flip, and center rotation). The first contains approximately 11,000 images with approximately 20% of them containing smoke, the second contains 43,000 images with the same proportions as the first, and the third contains 17,000 images but is balanced. The importance of having a balanced data set is shown, indeed the best test results were obtained with the model trained on the third set. Third, the model is tested on two other datasets and showed it had a similar or better detection rate (95% on the first dataset and 96% on the second) than popular CNNs like AlexNet and VGG16 and a slightly better accuracy rate (97.8% and 98%) but with three times fewer parameters than AlexNet and six times fewer parameters than VGG-16. Finally, the model has a slightly lower detection rate than [42] (which has 98% and 99.5% on the test sets), a similar accuracy rate but has a much lower false alarm rate (seven times lower on the first test set and four times lower on the second test).

In 2020, [45] proposed a deep dual-channel neural network using two different subnetworks for feature extraction. The first network is called the Selective-based batch normalization network (SBCNN) and is inspired by [44], it is composed of six convolutional layers, three max-pooling layers, and three fully connected layers. The second sub-network is called Skip connection-based neural network (SCNN), it is composed of 11 convolution layers, two max-pooling layers, seven normalization layers, and one global average pooling layer instead of fully connected layers, and implements skip connections to address the vanishing gradient problem. Both sub-networks are trained similarly to [44], using mini-batch SGD, with the dropout technique, momentum trick, and learning rate decay.

Input images are divided into sub-images, each sub-image is labeled as whether it contains smoke or not. The performance of each subnetwork is evaluated individually at first, on the same test sets as [44]. On the two test sets, both subnetworks have very similar detection rates, accuracy rates, and false alarm rates, and detection rates are better than [44] but false alarm rates are higher. Afterward, the two subnetworks are merged, the three fully connected layers of the first and the average global pooling layer of the second are removed, and only the parts responsible for feature extraction are kept and concatenated. A convolution layer followed by a global Average Pooling is added for classification. The final architecture shows excellent results on both test sets with accuracy rates of 99.7% and 99.4%, detection rates of 99.5% and 99%, and false alarm rates of 0.12% and 0.24%. For these three metrics, the proposed model performs better than AlexNet, VGG, GoogleNet, ResNet, and DNCNN [44]. This is done using only 7 million parameters, which is approximately three times less than DNCNN which is the model with the fewest parameters among those compared.

3.4.3 Discussion

Methane shares many properties with smoke, both are non-rigid, have irregular shapes, are influenced by wind direction and speed, dissolve over time, are in motion, fluctuate over time, and are emitted from a particular source [46]. However, smoke is visible, meaning it can be seen and captured by a regular RGB camera, while methane is invisible and cannot be captured by a regular camera, it is usually retrieved in grey scale [46].

From our literature review, we can affirm that smoke in images and videos is generally detectable with good accuracy, whether with traditional features [40, 42, 41] or with CNNs [44, 43, 45]. Smoke appears to be characterized by its colour [39, 40, 41], saturation [39, 40], edges [40, 44, 43, 45, 41] and textures [44, 43, 45, 42, 41]. CNNs [44, 43, 45] are able to identify and recognize smoke emissions with great accuracy.

However, the presented results are obtained on problems that differ from ours in several

points. First, they all use images taken from the front, while satellite images are taken from above. Secondly, the distances between the smoke and the camera lens seem very small, a few meters, while for our problem the images are taken at average distances of 500 km. Third, when there is smoke, it represents a very large proportion of the image and covers almost its entirety, and it is very easily distinguishable by the human eye. This is not the case for our problem, where the methane plumes represent a very small proportion of the image and can even be difficult to detect with the human eyes in certain cases. Finally, the datasets used either present images where there is clearly smoke or images which do not present smoke at all, in our case, there may be the presence of CH₄ in the atmosphere without this being linked to a plume.

3.5 Automatic methane plume detection

3.5.1 Methane leak detection in infrared camera

In 2020, as part of a controlled release experiment, the **GasVid dataset** is built [46]. It consists of 31 infrared videos of 15 frames per second for 24 minutes, recording methane leaks at different distances and for different source rates. The videos were recorded over a period of 4 days, and 2 different chemical separators and a tank are considered as emissions sources. Each frame is labeled by its distance from the camera and leakage rate, resulting in a dataset of approximately 1 million grayscale frames of 240×320 pixels. Each video is a combination of the distance from the camera (4.6 m, 6.9 m, 9.8 m, 12.6 m, and 15.6 m), the angle and orientation of the camera, 8 different source emission rate classes are recorded, ranging from 5.3 to 2051.6 g CH₄/h and including a no-leak category.

About 117400 frames of the GasVid dataset are used to make **three experiments** [46]. Three CNN architectures are proposed: GasNet-1, 2, and 3. They present gradual levels of complexity and are based on the *conv-pool* structure which is composed of a convolution layer, a batch normalization layer, and a maximum pooling layer. RELU activation function, dropout regularization and Adam optimizer are used during training. GasNet-1 presents one conv-pool structure followed by one hidden layer, GasNet-2 presents two conv-pool structures followed by two hidden layers and GasNet-3 presents fourth Conv-Pool structures and two hidden layers, all these architectures use the softmax function for prediction (leak vs no-leak). For all experiments, the data from one separator is used for training and validation (60%), while data from the other one is used for testing (40%) and image pixels are normalized.

The first experiment consists of assessing the importance of background subtraction on detection performances while keeping model complexity constant, to do this GasVid-2 is trained

without background subtraction and with 3 different background subtraction methods:

1. fixed background subtraction: using the average pixel value of all the frames from images without leak as the background threshold.
2. moving average background subtraction: generating a background threshold for a frame using the median of previous frames.
3. Gaussian-based background subtraction: using a Gaussian distribution of pixels to identify background pixels.

Without background subtraction, for the shortest distance (4.6m) the best detection accuracy is 82% for the highest source rate. However, the performance degraded quickly for an increased distance and lower source rates, until obtaining random accuracy. For the fixed background subtraction, detection accuracies are higher than 93% for all source rates for short distances (4.6m and 6.9m), however for higher distances the model has trouble differentiating between frames with low emission rates and those without a leak. Generally speaking, accuracy increases with the magnitude of the emission rate, except for the highest distance where accuracy is low. For moving average background subtraction, accuracies of detection are higher than 97% for all leak sizes and for short distances (4.6m and 6.9 m), but decrease for higher distances. However, detection accuracies are still higher than 90% for high leak rates. Gaussian-based background subtraction presents a similar behavior for short distances (lower than 10m) but shows worse performances for other distances. This experiment highlights the importance of background and noise removal before passing the data to a CNN and that the emission rate and the distance from the lens are very important factors in leak detection.

The second experiment consists of assessing the performances of the three CNN architectures using moving average background subtraction. The results showed that for small distances, the three models present very similar results. However, for higher distances, GasNet-2 performs better than the two other models. This can be explained by the fact that for small distances, recognition is quite simple and that all models can achieve it. For larger distances, GasNet-1 is not complex enough and GasNet-3 is too complex and may require more training data to show good performances.

The third experiment consists of testing different model aggregation methods. To do that, GasNet-2 architecture and moving average background subtraction are used and 3 methods of aggregation are compared:

1. train one model for each distance and leak size combination, this results in 35 trained models.
2. train one model for each distance for all leak sizes, this results in 5 models.
3. train only one model for all combinations of distances and leak sizes.

The first method showed the worst results for all distances, with an average detection accuracy of 93%, this can be explained by the fact that the individual models did not have enough data to train effectively compared to those from other methods. Methods 2 and 3 present similar results, with respective average detection accuracies of 94% and 95%. Method 2 performs slightly better for shorter distances while method 3 performs better for longer distances, especially for the longest one, where method 2 has an accuracy of 86% while method 3 has an accuracy of 91%. This can be explained by the higher size of the training data for method 3.

In 2022, [7] extended the work of [46] on the classification of the GasVid dataset's frames, by including the temporal aspect more rigorously in the data provided to the models, such that each input is of dimension $240 \times 320 \times 15$, this means that each observation consists of 15 consecutive video frames, equivalent to 1 second of video from the dataset. To do this, the same preprocessing as [46] (background subtraction by moving average and pixel normalization) is used, and two models are proposed. The first is a 3D CNN, which is simply a CNN that takes as input several channels or images (15 in this case), which has the same architecture as GasVid-3 [46] except that it considers different input dimensions (which is reflected in the dimensions of the convolution kernels). The second implements the *Convolutional Long short-term memory* (ConvLSTM) architecture, which combines convolution architecture and recurrent neural networks that are used to learn temporal dependencies in the data, it is therefore designed to extract visual features and temporal dependencies. The 2 models are compared with the mode value of the predictions of GasNet [46] over 15 frames, on short-distance (4.6 m and 6.9 m) images only, for three problems :

1. 8 classes classification problem: prediction of the leak rate category proposed in [46], from no leaks to the highest level
2. binary classification problem: whether an input contains a leak or not
3. 3 classes classification problem: prediction of the leak rate category between 3 classes (small, medium, large)

The results show that 3D CNN and ConvLSTM perform better than GasNet [46]. Both models showed very similar performances, with a slight advantage for the 3D CNN. This shows that incorporating the temporal aspect of a methane leak can improve detection accuracies.

3.5.2 Airborne and satellite retrievals

In 2020, [21] proposed a CNN named *MethaNet* to detect plumes or quantify their rates of emission from generated AVIRIS-NG data. A dataset is created by collecting 3000 background scenes from different landscapes, by sampling AVIRIS-NG retrievals. The 3D Large Eddy Simulation method is used to generate realistic 2-D CH₄ images. 7000 plumes were created with various emission rates varying between 0 and 2000 kg/h and different wind speeds (1 to 8 m/s). To generate more data, plumes, and background scenes are split into 3 sets (train, validation, test) and are augmented independently by rotations, translations, and scaling for plumes. Afterward, the plumes and backgrounds are merged to create over 293,000 300×300 CH₄ images, covering a range of 1.5×1.5 km².

MethaNet is composed of four convolution layers, three max-pooling layers, 1 dropout layer, and two fully connected layers. RELU activation function is used for the regression task (emission rates predictions), while for the classification task (whether there is a plume or not) it is a softmax layer.

For the regression task, the model demonstrates mean absolute error percentages of 22% and 17% across all images and for plumes with a magnitude exceeding 40 kg/h, respectively. The model is also tested with a generated Gaussian distributed random noise (instead of the background scenes) with CH₄ concentration levels corresponding to the lower-end concentration levels that can be found in the real backgrounds and obtained mean absolute error percentages of 17% and 16% across all images and for plumes with a magnitude exceeding 40 kg/h, respectively. These results show first that the model is capable of capturing the shape of the plumes and their texture to derive fairly precise rate estimates. Second, plumes with smaller emission rates are more difficult for the model to understand. Third, the amount of noise in the image affects greatly the quality of the prediction.

For the classification task, the plumes are removed from 50 % of the images in the augmented dataset. The model obtained detection accuracies of 90% and above 50% respectively for emissions rates above 100 kg/h and around 50 kg/h. By replacing the backgrounds with generated noises as for the regression task, model performances improved for plumes under 100 kg/h, and the detection rate for emissions rates between 40 and 75 kg/h improved by approximately 15%. Similar conclusions to the ones from the regression tasks can be drawn.

In 2020 [48] proposed two methods to segment plumes on an augmented dataset derived from 46 AVIRIS retrievals of 22000×1400 pixels. The first one uses a preprocessed CH₄ channel as well as 3 RGB channels with a Mask-R-CNN. Retrieval patches are generated with a sliding window algorithm over the retrievals and are used as the network inputs. The best results of 0.832 for intersection over union and 0.92 for f1-score for the plume detection are obtained with a window size of 256×256 and 50% overlap. Results decrease with increasing window size.

The second, Hyperspectral Mask-RCNN takes as inputs 256×256 patches of 432 hyperspectral bands instead of the first method's four channels. A match filter algorithm is applied via a sliding window on the spectral dimension, producing a methane image for each set of bands in a window. A Mask-R-CNN for each set of bands is trained to recognize the CH₄ appearance retrieved from different hyperspectral bands. The predicted masks by each network are merged by a 2 layer perceptron to produce the final output. Compared to the first method, the intersection over union score improved to 0.85 but the f1-score remained almost the same.

In 2022, [47] used a U-NET architecture to segment plume pixels from PRISMA retrievals and also adapted this architecture to the classification task. A *matched filter* algorithm is used to obtain methane enhancement images from 40 PRISMA's hyperspectral retrievals for a total of 75 plumes. To augment the dataset, a plume transfer method from Sentinel-2 images to PRISMA is developed. This way, 1000 Sentinel-2 plumes were adapted to the PRISMA instrument and merged with PRISMA backgrounds at random positions and orientations. The model obtained an intersection over union score of 0.61 for segmentation, which is considered a correct score. For classification, a precision of 0.88 and a low recall of 0.42 are obtained for plume detection.

In 2023 [16] proposed a solution for plumes detection on 32×32 Tropomi CH₄ retrievals based on two-chain classifiers. The first is a CNN composed of four convolutional layers, two max-pooling layers, two fully connected layers and using binary cross-entropy, Adam optimizer, dropout regularization, and RELU activation. The second is a SVM that takes as input 40 handcrafted features and the CNN prediction probability. The features are mainly derived from CH₄ pixels including thresholding, statistical properties of pixels like the Skewness of the distributions of the pixels for each channel, wind and aerosol optical thickness data, and pixels correlation between different channels using *Pearson correlation coefficient*.

The second classifier is only used if the first predicts the retrieval as containing a plume to differentiate artifacts from real plumes. Small datasets from 2019 and 2020 are augmented to train and test both models, resulting in respective precision and recall of 91 and 93% for plumes. An unlabelled 2021 dataset is used to assess the method performances, out of 26

444 scenes 3.3% are labeled as containing plumes, and after manual verification between 61 and 76% of them seem like real ones. The plume detection rate is not evaluated because it would imply manually going through all 26,444 scenes.

3.5.3 Discussion

According to our literature review, CNN methods are the first choice for CH₄ plume recognition. Indeed, [16] is the only approach that uses hand-crafted features, but they are used following a CNN to improve its predictions.

[46, 7, 21, 16] showed that the emissions rate of a plume is an important factor in its detection. All, the models are trained out of several thousand images, and by using data augmentation when necessary. Showing the importance of having a large dataset and that the lack of labeled data taken from the sky [52] still seems to persist. [21, 46] highlights the impact that the amount of noise can have on detections for mid-rate leaks in plume recognition. [48] supports the relatively small input sizes used by all approaches by demonstrating that using small inputs improves predictions.

In [46, 7, 21], the detection rates are high (above 85%), and in the [48] the f1-score (0.92) for the plumes is very high. [16] approach showed high precision (0.91) and recall (0.93) for plumes on the augmented test sets but expressed more modest results on the non-augmented test set with an estimated precision between 0.61 and 0.76 and an unknown detection rate. On the other hand [47] showed a low detection rate of 42% but a high precision of 88% and a correct segmentation score for the detected plumes, this means that when the model detects a plume it is able to segment it well, but it is not capable of detecting several plumes.

[47, 16] highlighted that methane emission detection can be inaccurate due to aerosol, natural variability, or surface with a spectral signature similar to methane, in such a way that it is possible to have surfaces appearing with high intensity of methane enhancement but which are not plumes. This is a recurrent problem with hyperspectral data [49, 52] and this could be found in GHGSAT's retrievals.

Several considerations must be taken into account for [46, 7] results. First, they are obtained on plumes from an infrared camera, having a higher CH₄ detection precision but a very low special coverage (tens of meters) compared to our data, the retrievals are taken from the front while GHGSAT's from the sky, the cameras are made to detect small plumes on a small scale while satellite data are made to detect large emissions. Second, the experimental conditions do not take into account the diversity of cases found in the GHGSAT data. GasVid dataset purposely excluded elements that could disturb the quality of the frames such as

people, cars, vapor, steam, and moving vegetation. All GasVid images capture two different instruments at different angles and distances, which does not allow testing the results for different surroundings and atmospheres, which can influence the acquisition of hyperspectral data [52].

In addition, the dataset was recorded in just four days, which does not allow it to take into account the effect of different seasons, which may also influence this type of acquisition. Third, the best results proposed by [46] use the temporal aspect of frames by their background extraction method. Similarly, [7] naturally exploits this property since the proposed work is based on temporal CNN networks.

The results of [21, 47, 48, 16] can be considered as a point of comparison for our work, they are data taken from the sky, presenting relatively large coverage areas, with similar emission rate scale than GHGSAT’s retrievals, except for [16] where they are much higher. However, it should be taken into account that the first two methods used realistic plume simulators for their experiments, this could impact the amount of noise present in their retrievals compared to GHGSAT’s. This allows for a simple data augmentation process and control of the experimental conditions. For example, [21] reduced the number of variables in the problem by controlling wind direction and speed during plume generation and generated several plumes within a defined leak rate range. Furthermore, [21, 47] used data augmentation that showed very good results on augmented sets but less good results on a non-augmented test set for [16]. Finally, [48] exploit RGB channels or hyperspectral bands, and [16] use wind and aerosol data to enhance predictions, these types of data are not included in the GHGSAT dataset.

CHAPTER 4 METHODOLOGY

Our goal is to distinguish between methane satellite retrieval that contains plumes (**class 1**) from those that don't (**class 0**). We define a **plume retrieval** as a retrieval that presents at least one plume.

An approach based on the generation of new channels and the extraction of handcrafted features with the use of traditional machine learning models was designed from observations and developed.

The proposed solution is described in this section. First, an overview of the approach is given. Second, the dataset is presented and studied. Third, the created channels and the handcrafted features are explained. Fourth, the feature selection and model fine-tuning processes are described. Fifth, the simple informative comparison experiment that was done to compare the image-based feature to CNNs is described. Six, the method for evaluating the approach is presented, to finally ends with a description of the detection pipeline that implements the approach.

4.1 Introduction and justification of the approach

4.1.1 Overview of the approach

With the aim of **developing a prototype method** to evaluate the feasibility of retrieval classification, a handcrafted feature approach is proposed and evaluated. It involves defining and extracting handcrafted features based on the knowledge acquired through data observation and analysis, from retrievals, and training multiple machine learning models with these features to achieve the classification task. This approach works with inputs of different sizes.

The proposed features fall into two categories, **image-based** and **geolocation features**. The image-based features are features extracted on the pixel values of several generated image channels, in addition to the *CH₄-SER* and *CH₄-C* provided channels. Geolocation features are defined based on the geolocation data present in the retrievals and are not related to the visual appearance.

The extracted features, preprocessing, and dimensionality reduction methods are first analyzed through preliminary classification experiments with two models. Furthermore, two CNNs are trained and compared to our image-based features through a simple experiment to assess their pertinence compared to the dominant approach. Subsequently, other models

are trained and fine-tuned with a Random Search.

The approach is evaluated on 5 different splits (train/test) of the dataset that preserve the class and proposed categorization proportions (subsection 4.2.6). They are used to evaluate the general classification performances and the plume retrieval detection rate in each emission rate category.

4.1.2 Justification

The choice between the handcrafted feature methods or the most recent CNN approaches was examined for the development of the prototype approach. Both methods satisfy the deep learning requirement (section 1.4). Indeed, handcrafted features can be used as inputs for a traditional deep NN, while CNNs are by nature deep learning approaches.

While all the presented approaches on methane plume detection except [16] rely only on CNNs, many handcrafted feature methods presented good results in smoke detection [40, 41] and although they are not state-of-the-art they displayed good results in ILSVRC challenges [57, 58].

What mainly steered us towards this approach instead of a CNN is that the dataset presents retrievals of varying dimensions and aspect ratio which requires additional preprocessing for using a CNN which requires the inputs to be of **the same size**.

Background padding could be added to all the images so that they have the maximum number of rows and columns present in the dataset. It is not practical because it would make the retrievals much larger, resulting in a lot of computing power and memory required, more training time, and more complex classifications. After cropping, the maximum number of rows and columns are 4054 and 3169 (see subsection 4.2.3). This idea is supported by the fact that all the presented approaches on methane plume detection [48, 16, 21, 46, 7] used image inputs smaller than 300×320 pixels, and [48] showed that the detection performance decreases as the dimensions of the input increases.

Similarly, a downsampling approach could be used to reduce the size of retrievals, but it implies additional preprocessing and parameterization. This can impact the final result and introduce information loss. If we decided to use the smallest dimension as the downsampling dimension, the largest images would suffer a great loss of information and the shape of the retrievals would be strongly distorted, knowing that the dataset presents different aspect ratios.

Another solution could be to separate each image into several sub-images (patches) of the same size and use them as a working dataset for a CNN. This approach is excluded due to the

scope of the work (section 1.4). Furthermore, it would also have involved several variables and considerations including the choice of the size of the patches, the use or not of overlap between the patches, the labeling of cases where plumes are distributed across many patches, and the number of patches to use. In addition, the resulting dataset would be extremely class-unbalanced. Likewise, because of the scope of the work, the use of object localization methods (subsection 3.2.3) is excluded.

Additionally, handcrafted feature methods can present advantages over CNNs. First, they are interpretable, they provide an explanation of the information that allows models to reason, which is not the case with CNN approaches where features are learned automatically. Second, computational requirements for training models are reduced with handcrafted features because the model takes the features as input instead of taking the entire image. Third, they are generally less dependent on the chosen model, its optimization, and the dataset size compared to CNNs, because features already present meaningful information.

Moreover, through visual observations and analysis of the data (section 4.2), handcrafted feature solutions seemed to be able to produce good results.

For all this reason, the presented approach was chosen for CH4 retrievals classification.

4.2 Dataset analysis

Dataset analysis is a crucial step in order to propose an adequate solution to the problem, allowing the understanding and familiarization with the dataset.

First, the dataset and the visual insights obtained by working and analyzing the images are presented followed by an analysis of the dataset for several variables: retrieval size, class balance, plumes, satellites, emission rates, geolocation, channel values correlation, and pixel values.

4.2.1 Dataset presentation

Dataset format

The raw dataset as provided by GHGSAT consists of the following:

1. A CSV file where each row contains at least the following information: retrieval ID, acquisition time, satellite ID, and the paths to various image channels. If the image contains one or more plumes, the following information is provided for each plume: a path to an image containing the binary mask of the plume, the source emission rate in

CH₄ kg/h, the industrial sector of the source, the direction and wind speed, and the position of the source origin in latitude and longitude.

2. A folder containing the different image channels for each retrieval listed in the CSV file.

It contains 16779 retrievals, acquired between 2020 and 2023 from GHGSAT's satellites C1, C2, C3, C4 and C5.

Retrieval

Each retrieval comprises three images or channels of the same size, representing different information of the same scene:

1. *CH₄-C* channel: each pixel represents the methane concentration in a vertical column in mol m⁻².
2. *CH₄-SER* channel: which is obtained by dividing each pixel of *CH₄-C* by the associated instrument error. $CH_4-SER = \frac{CH_4-C}{ERROR}$. A pixel with a high associated error leads to a small value in this channel. It takes into consideration the error of the instrument to adjust the retrieval and remove a part of the noises.
3. *ALBEDO* channel: albedo is a measure of the reflectivity of a given surface. A high albedo value indicates that the surface reflects a lot of light, as opposed to a low albedo value, which corresponds to a surface that does not reflect well.

Additionally, each image or channel within a retrieval is **georeferenced**, using the GeoTiff [70] format. This format includes metadata that allows precise mapping of pixels to Earth locations.

Exclusion of wind data

The wind data was excluded from the work for 3 reasons. First, they are only provided for the plumes. To include them, it would be necessary to do data collection work such as: mapping the position of the retrievals and using it with their acquisition time to fetch wind data from geological databases. Second, these types of databases have low temporal and geographic precisions, which could induce a lot of noise [25]. Third, the literature indicates that its use is not necessary for the detection of the methane plume [46, 7, 21, 48].

4.2.2 Visual aspect and observations

Visual insights obtained from working with the data are presented.

Problematic cases in the context of binary classification

Two cases of retrievals that may lead to difficult predictions are identified. The first consists of retrievals that display large values of CH_4 but which do not contain plumes, this case can cause false alarms (FP) and reduce precision. Second, some plumes are very small and have very low intensities, which makes it difficult to differentiate them from their surroundings. These cases can cause a drop in the image detection rate (FN).

Figure 4.1 presents in each row CH_4 - SER retrievals from four categories : (1) No Plume and Low CH_4 - SER Level, (2) No Plume and High CH_4 - SER Level, (3) Hard-to-Detect Plumes, and (4) Easy-to-Detect Plumes.

Channels

By plotting and studying the three different channels CH_4 - SER , CH_4 - C and $ALBEDO$ with the masks when present, two main observations are made.

First, CH_4 - SER channel permitted to see most of the plumes visually. In contrast, CH_4 - C presents the information but in a much less precise, irregular, and noisy manner than CH_4 - SER .

Second, there is no evident visual correlation between the $ALBEDO$ channel (also using it simultaneously with other channels) and the presence or absence of plumes.

Figure 4.2 illustrate retrievals for channels CH_4 - SER , CH_4 , $ALBEDO$ and CH_4 - SER - CI (see section 4.3), each row correspond to a retrieval and each column to a channel.

We can observe that the channel that presents the most information is CH_4 - SER and that the $ALBEDO$ channel does not present obvious information regarding the presence or absence of plumes. The second row highlights the difference between the two CH_4 channels, where the plumes are hardly visible in CH_4 - C .

Plumes

Plumes are generally associated with regions of high pixel values of CH_4 - SER but do not necessarily contain all the highest-value pixels present in the retrieval. They are generally elongated but not always, their shape, intensity, positioning, and size vary a lot from one to

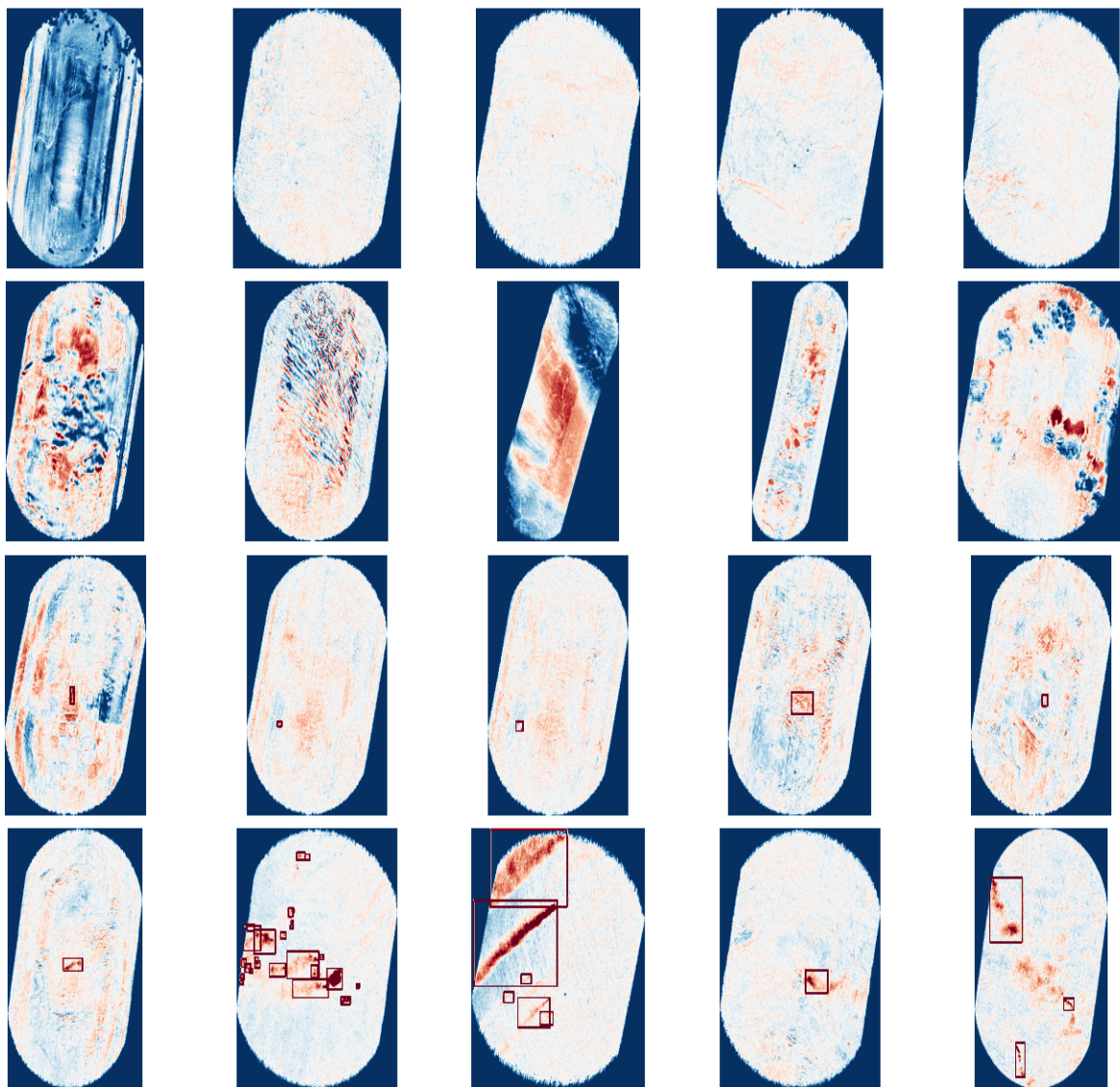


Figure 4.1 CH_4 -SER retrievals from four categories, from top-to-bottom : (1) No Plume and Low CH_4 -SER Level, (2) No Plume and High CH_4 -SER Level, (3) Hard-to-Detect Plumes, and (4) Easy-to-Detect Plumes. Plumes are framed. Color scheme is applied, where dark blue and dark red represent respectively very low and very high CH_4 concentrations.

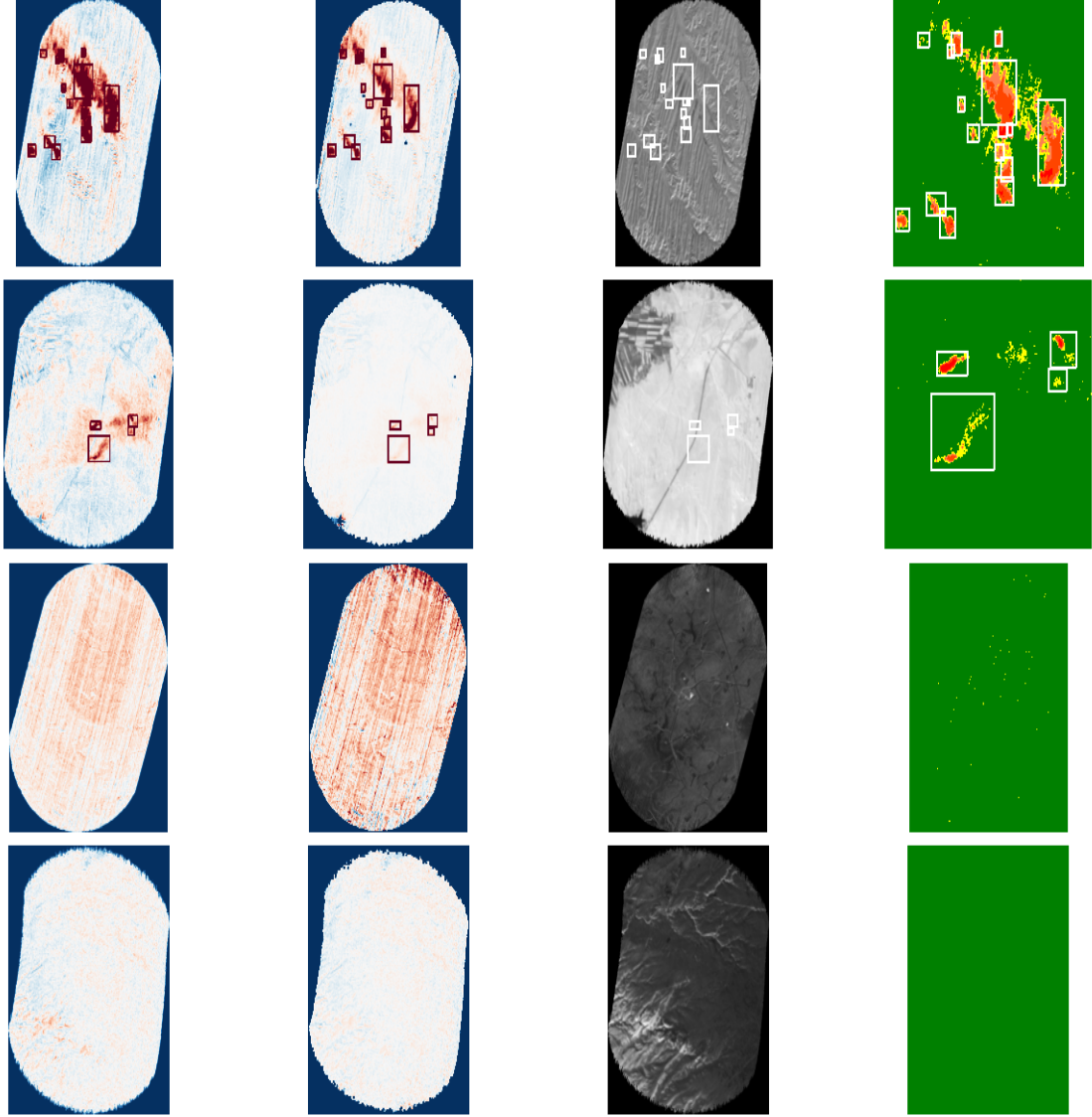


Figure 4.2 Visual comparison of channels. Each row corresponds to a retrieval and each column to a channel, from left to right: CH_4-SER , CH_4 , $ALBEDO$ and $CH_4-SER-CI$. For, $CH_4-SER-CI$, each of the five defined intensities is associated with a specific color, ranging from lowest to highest intensity: green, yellow, orange, red, and dark red. Light value of $ALBEDO$ indicates high light reflectance, while dark value indicates low reflectance.

another.

The fourth row of figure 4.1 shows fairly easy-to-visualize plumes, while the third shows plumes that are really hard to detect even with the presence of bounding boxes. From observations, 2 major cases of these complex plumes are identified: the plume is very small or the intensities of the plume’s pixels are indistinguishable from its surroundings.

Artefacts

Generally speaking, the images are quite noisy, it is also possible that several high-intensity pixels are present in a grouping (like what would be expected for a plume) but that it is not a plume, but some other artifact. This issue is recurrent with aerial and satellite methods according to the presented works.

If sensor failures are omitted, this could be explainable either by the presence of an environment where the methane rate is very high like aerosols [47, 16], without being necessarily a plume. Or by a set of events which under certain conditions cause some materials to have spectral signatures close to that of methane, thus distorting the data resulting from hyperspectral processes [47, 52, 49].

Some images contain a lot of artifacts that are not plumes and others do not. Artifacts vary in size, shape, and intensity. The first two rows of figure 4.1 present respectively retrievals without artifacts and with artifacts.

4.2.3 Image size and cropping

In order to get information about the size of the images in the dataset, the minimum, maximum, and average of: the number of rows, columns, and pixels in an image are computed before and after cropping.

Cropping consists of removing empty rows and columns associated with padding added during the retrieval process. Figure 4.3 shows retrievals of different shapes before and after cropping and table 4.1 illustrates statistics on the numbers of rows, columns, and pixels in a retrieval before and after applying cropping.

Retrievals vary greatly in size and in aspect ratio and tend to have more rows than columns. Cropping reduces the average number of pixels by 3, giving an average number of 322 507 by retrieval.

Table 4.1 Statistics on the numbers of rows, columns, and pixels in a retrieval before and after applying cropping

	Rows			Cols			Pixels		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Before Cropping	613	7191	1107	594	5785	910	493 885	41 599 935	1 039 547
After Cropping	262	4054	636	166	3169	483	63 080	12 847 126	322 507

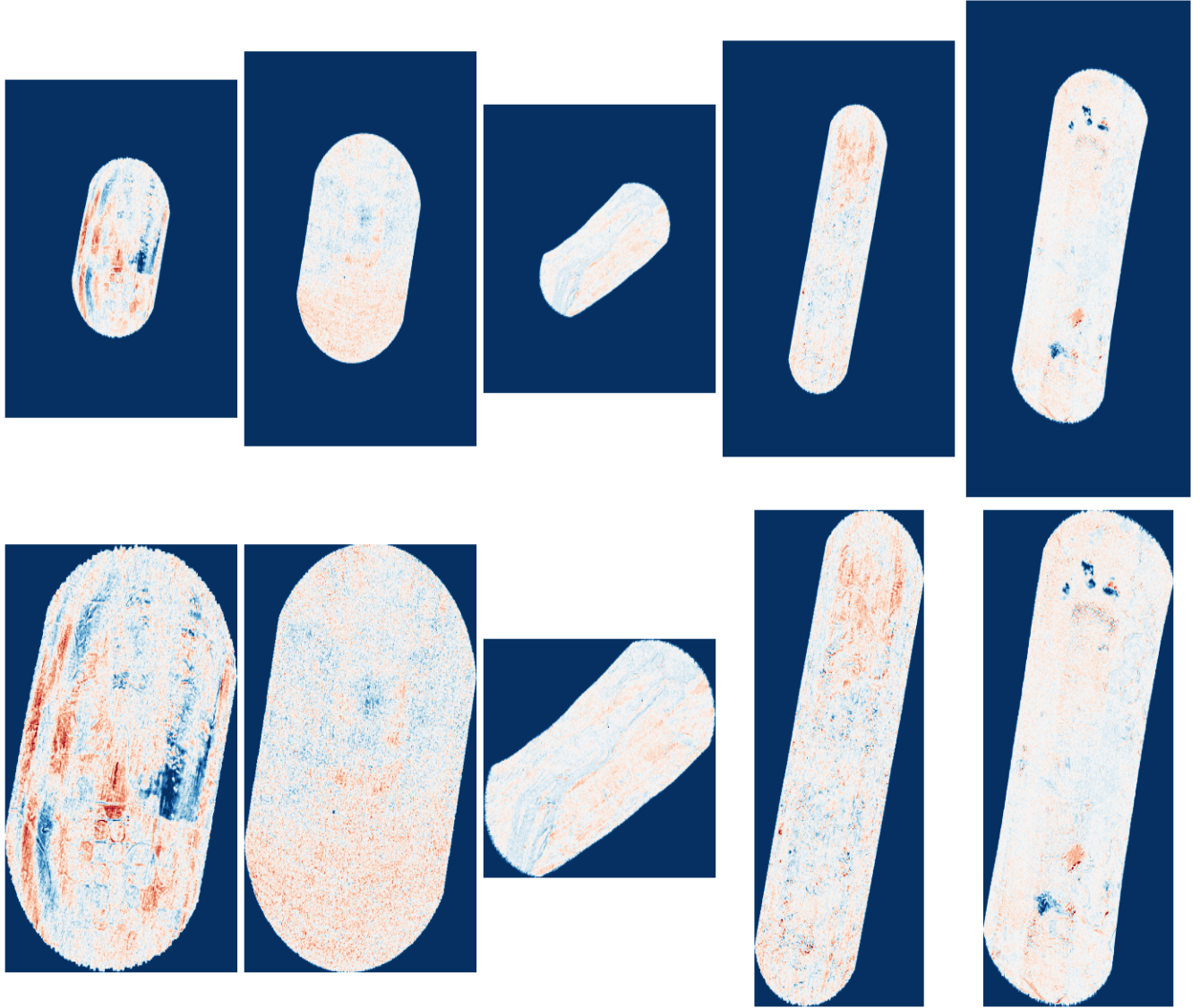


Figure 4.3 Cropping effects for different shapes of retrievals. Both rows represent the same retrieval before and after cropping. The dimensions, in pixels, before and after cropping are indicated from left to right: (895x842)-(478x356); (1040x838)-(603x448); (1402x1553)-(657x879); (1522x1165)-(1059x496); and (2253x1398)-(1609x846).

4.2.4 Class balance and plumes

The dataset of 16779 retrievals contains 4315 plume retrievals. that's almost **26%** of the dataset. This indicates that the dataset is not balanced.

The total number of plumes in the dataset is 9756. This indicates that excluding images that do not contain plumes, the average number of plumes per retrievals is 2.26. The retrieval with the most plumes has 35 and 91% of retrievals that contain plumes contain between 1 and 5. Each plume in the dataset is unique, preventing any duplications across retrievals.

Moreover, 57% of plumes come from the oil and gas industry, 26% from mining, and 15% from waste management. The remaining 2% is distributed between different sectors. Similarly to [16] the principal sector of emission is the oil and gas industry.

4.2.5 Satellites exploration

The retrievals are acquired by five distinct satellites (C1-C5), each operating independently to ensure there is no overlap in coverage, each satellite is responsible for capturing specific regions on Earth. Therefore, the objective of this analysis is to study the retrievals by satellite.

Proportion of retrievals by satellite

The proportion of retrievals acquired by each satellite is computed, allowing us to determine if one satellite produces significantly more acquisitions than others. Figure 4.4 presents the number of retrievals for each satellite.

C1 and C2 almost equally share 60% of the retrievals and C3, C4, and C5 equally share 40% of the remaining retrievals.

Number of plume retrievals and class balance

The number of plume retrievals for each satellite is computed. The goal is to determine whether some satellites produced more plume retrievals than others. Figure 4.5, present the number of retrievals with at least one plume for each satellite.

It shows that C2 captures more plume retrievals than C1 despite having almost the same number of retrievals, C2 alone contains 35% of retrievals with at least one plume.

The balance of classes within each satellite is also computed. C3, C4, and C5 have almost the same proportion of plume retrievals as the whole dataset, i.e. 27%,26%, and 27% respectively.

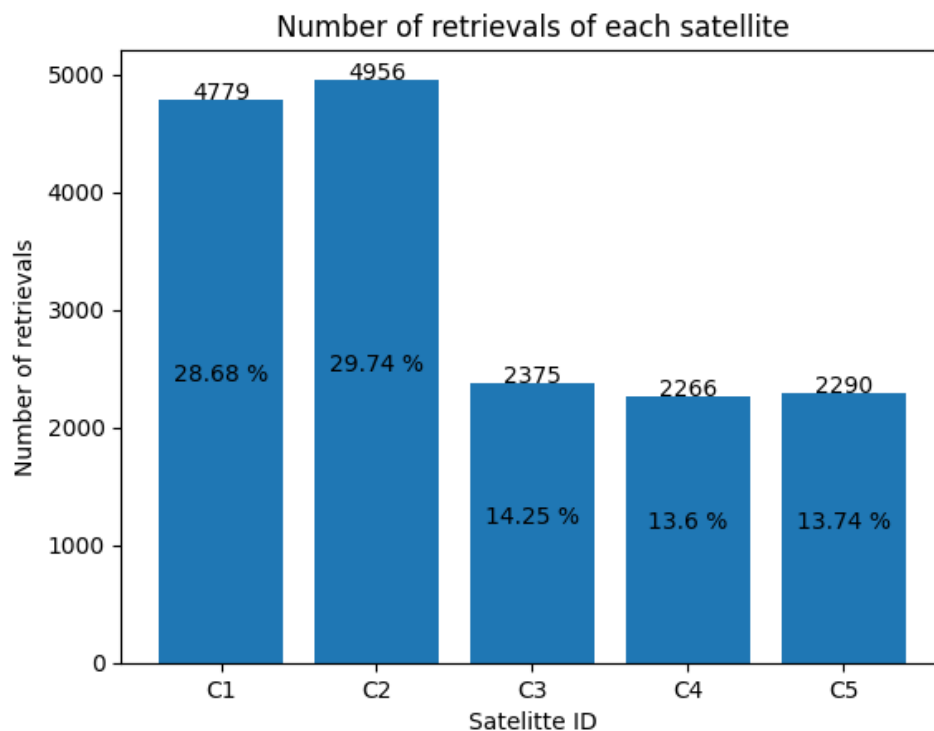


Figure 4.4 Number of retrievals of each satellite

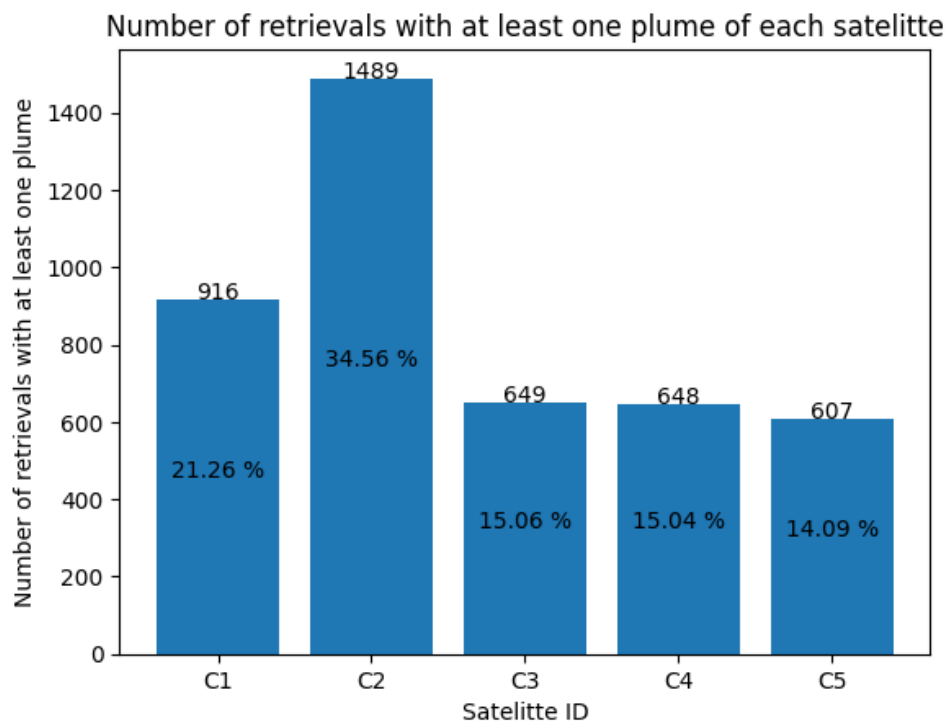


Figure 4.5 Number of retrievals with at least one plume for each satellite

While C2 has a slightly higher proportion with 30% as opposed to C1 with 19%.

4.2.6 Emission rates analysis and categorization

Since the works presented by [46, 7, 21, 16] showed that the source emission rate is an important factor in CH₄ detection, emission rates of retrievals are studied in order to categorize retrievals into different emission levels. This categorization is designed so that emissions with a low category are challenging to detect, while those with high categories are easy to detect.

Metric

Since a retrieval r can contain l plumes, we define the emission rate of a retrieval $s(r)$ as the sum of the emission rates of the plumes it contains :

$$s_r(r) = \sum_{i=1}^l s_r(p_i) \quad (4.1)$$

Where $s_r(p_i)$ is the emission rate of plume i .

Plume and retrieval emission rate analysis

In order to understand the magnitudes of the different emissions present in the dataset, emission rate statistics are computed for individual plumes. Table 4.2 presents several statistics on plume emission rates, namely: the average (Avg), the standard deviation (Std), several percentile values, and the maximum (Max).

Avg	Std	5%	15%	25%	50 %	75%	85%	95%	Max
2018	3437	226	387	540	1085	2178	3215	6363	88740

Table 4.2 Statistics of plumes CH₄ Emission Levels (in kg/h), rounded to the nearest Integer

[21] experiments with plumes between 0 and 2000 CH₄ kg/h, 75% of the plumes of GHGSAT retrievals are in a similar range (0-2178), 25% of plumes are larger.

Similarly, statistics on retrieval emission rates are computed. Table 4.3 presents the same statistics for retrieval emission rates.

Plume retrieval categorization

By visualizing the retrievals in each emission level quartile, the following was observed. Retrievals from the first quartile present minimal plumes and are hard to distinguish. From

Avg	Std	5%	15%	25%	50%	75%	85%	95%	Max
2932	4669	344	604	869	1710	3232	4637	9079	98014

Table 4.3 Statistics of retrievals CH_4 Emission Levels (in kg/h), rounded to the nearest Integer

the second quartile, they begin to be visible until the end of the 3 quartile, where the plumes are clearly visible. Certain retrievals between the second and third quartile can sometimes be very difficult to perceive. The plumes of the last quartile are very visible and generally large.

For this reason, the plume retrievals are divided into **3 categories** based on their emission rate :

1. the retrievals belonging to the first emission quartile are labeled as small
2. the retrievals belonging to the second and third quartile are labeled as average
3. the retrievals belonging to the fourth quartile are labeled as large

In this way, 50% of the plume retrievals belong to the average category and the small and large categories equally divide the remaining observations.

Figure 4.6 presents CH_4 - SER retrievals ordered by emission levels, with each row illustrating retrievals from different quartiles. The top and bottom rows represent respectively the lowest and highest 25% of retrievals based on their emissions.

Retrievals in the first row of Figure 4.6 fall into the small category, while the retrievals in the second and third rows can be categorized as average. Finally, the retrievals in the last row are representative of the large category.

4.2.7 Geolocation study

To try to exploit the hypothesis that certain geographic areas contain more emissions than others, for example, due to the concentration of emitting facilities, the retrievals are clustered according to their geodesic distance and the clusters are studied.

Methodology

A retrieval terrestrial position is defined as the latitude and longitude of its central pixel. This way, each retrieval is associated with a terrestrial position. This is done by extracting the terrestrial data encoded in GeoTiff format from the retrievals

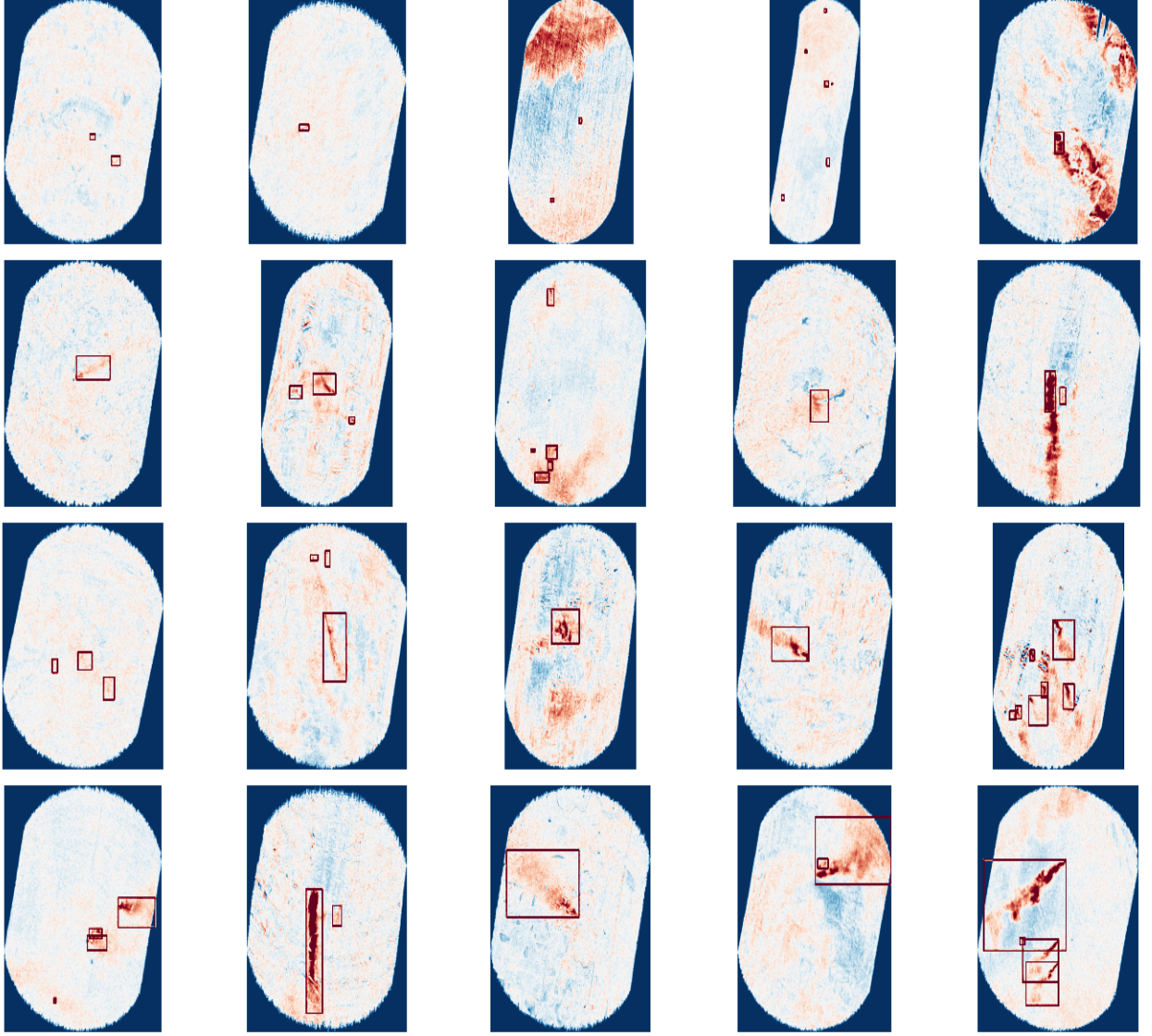


Figure 4.6 CH_4 - SER retrievals ordered according to their emission levels. Each row illustrates images from each quartile of emission levels, arranged in increasing order of emission level from left to right.

A simple *greedy clustering algorithm* to group retrievals into distinct clusters based on their position is implemented. A cluster is defined as a collection of retrievals and an associated position that represents the average position of the retrievals within it.

The algorithm involves iterating over unvisited retrievals and comparing the geodesic distance between their positions with that of the cluster. If the distance is below a certain threshold distance D , the retrieval is added to the cluster, marked as visited, and the average position of the cluster is updated. Algorithm 2 details this procedure.

Algorithm 2 Clustering Algorithm

```

1: procedure CLUSTERRETRIEVALS(retrievals, D)
2:   Input: A list of retrievals, a distance threshold  $D$ 
3:   Output: A set of clusters
4:   Shuffle all retrievals
5:   Initialize an empty set named visited
6:   Initialize an empty list of clusters clusters
7:   while there are unvisited retrievals do
8:     Take a random retrieval  $R$ 
9:     Mark  $R$  as visited
10:    Create a new cluster  $C$  with  $R$  as its only member
11:    Set the position of  $C$  as the position of  $R$ 
12:    for each unvisited retrieval  $U$  do
13:      if the distance between the position of  $C$  and  $U$  is less than  $D$  then
14:        Add  $U$  to cluster  $C$ 
15:        Mark  $U$  as visited
16:        Update the position of cluster  $C$  as the mean position of its retrievals
17:      end if
18:    end for
19:    Add cluster  $C$  to the list of clusters clusters
20:  end while
21:  Return: clusters
22: end procedure

```

Results

For an arbitrary distance of $D = 10$ km, 94% of the retrievals are found in clusters made up of at least two retrievals. Considering only clusters with at least two retrievals, the average number of observations contained per cluster is 7.3. The maximum number of observations found in a single cluster is 130.

By analyzing the observations contained within the same clusters, the following is observed. First, retrievals belonging to the same cluster are very similar, they generally present the

same artifacts, and shapes and have a similar background. Second, the clusters are quite homogeneous in terms of the presence of plumes. Indeed, clusters with several observations that all contain plumes or that all don't were found. Figure 4.7 present CH_4 -SER retrievals from different geographical clusters, each row presents retrievals within a cluster.

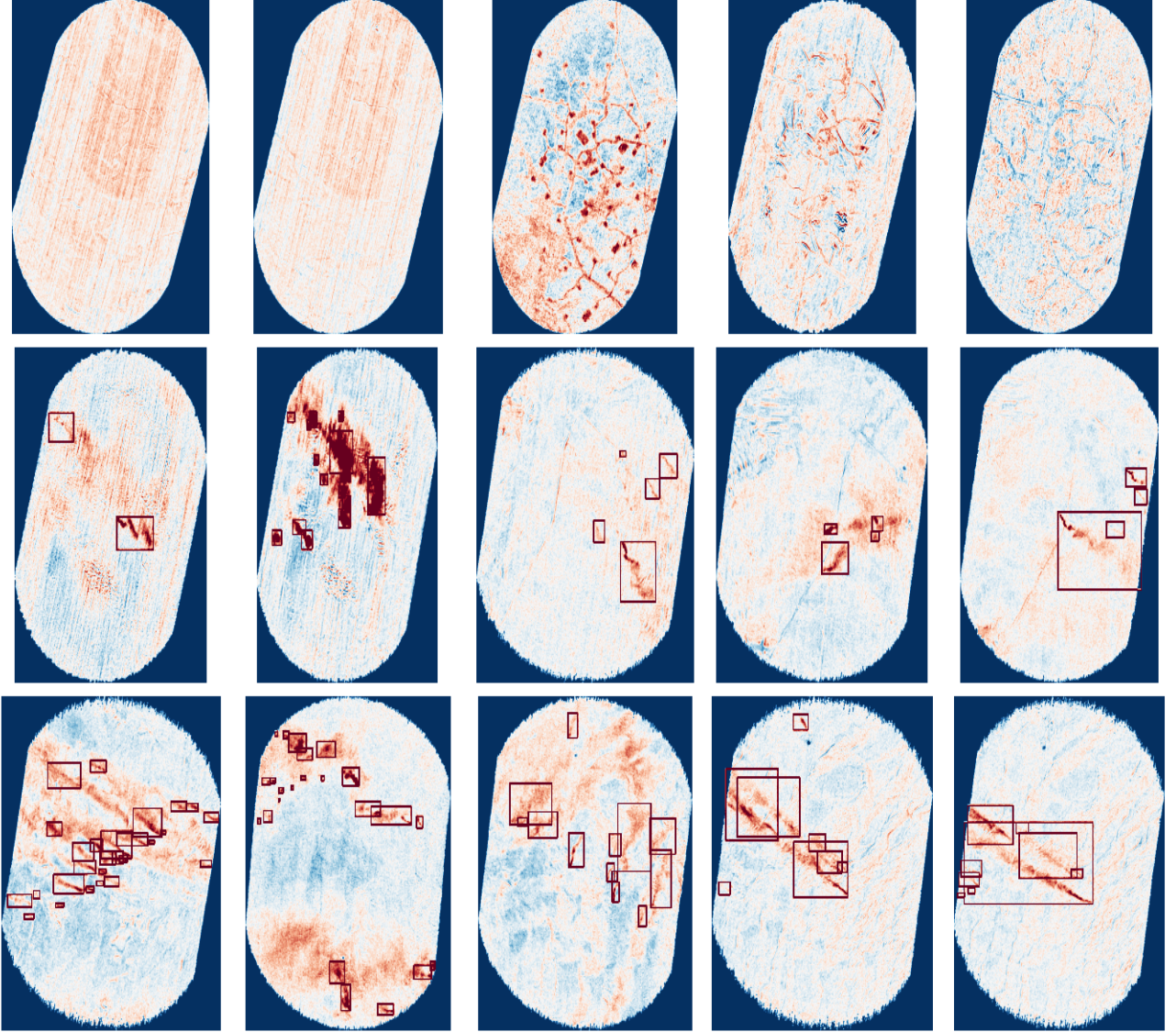


Figure 4.7 CH_4 -SER retrievals from different geographical clusters. Each row presents a cluster and retrievals that belong to it. The first row presents a cluster without plumes, and the other two present clusters with a lot of plumes.

These results are consistent with the retrieval process, the satellites pass through the same geographical areas several times. Certain areas often have concentrations of industries likely to emit emissions, while other areas do not.

4.2.8 CH₄ plume pixels values and proportions

The objective is to study the values of the pixels associated with the plumes and the proportion of pixels in a retrieval they present.

Values

The objective of this analysis is to compare the distribution of *CH₄-SER* pixel values inside and outside the plumes. To do this, first, the pixel values are rounded to the nearest integer. Second, using plumes masks, the frequencies of appearance of each value inside the plumes and outside the plumes are computed. Figures 4.8 and 4.9 respectively illustrate the frequencies of appearance of the different values inside and outside the plumes.

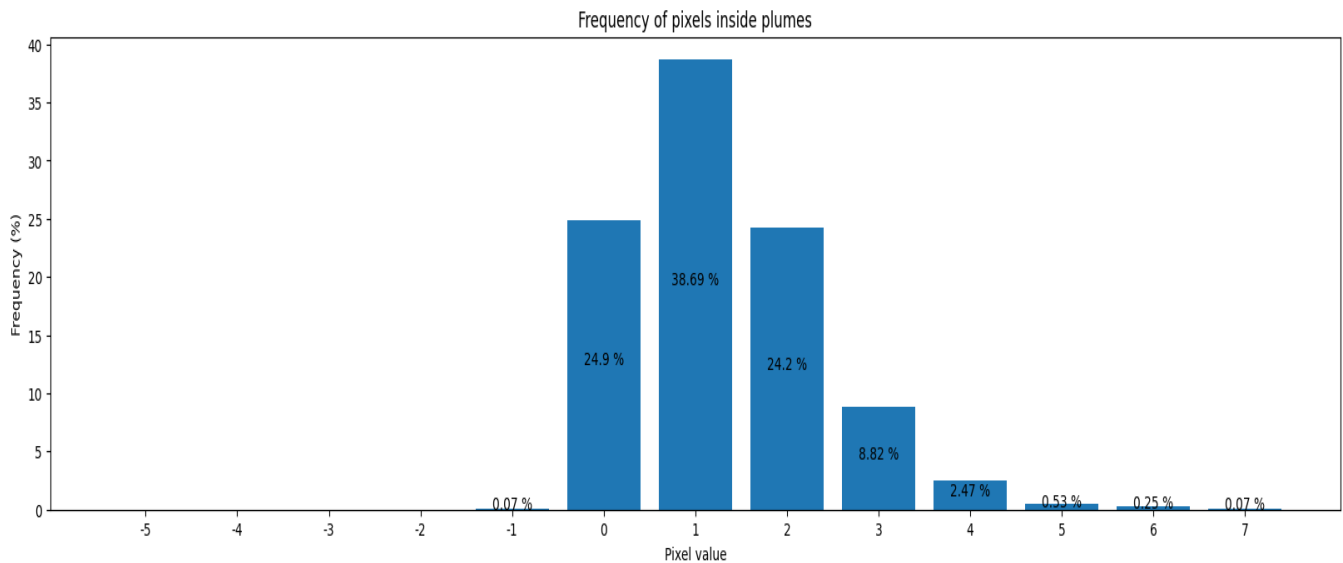


Figure 4.8 Frequencies of pixel values inside plumes for the *CH₄-SER* channel

Generally, all the values of *CH₄-SER* are between -5 and 7. Inside the plumes, very few values are less than 0 (0.07%) while outside the plumes, almost 11% of the values are less than 0. About 75% of the values inside the plumes are greater than or equal to 1, compared to only 15% outside the plumes.

These results show that the distribution of *CH₄-SER* values inside and outside the plumes is not the same, the plumes tend to have higher values.

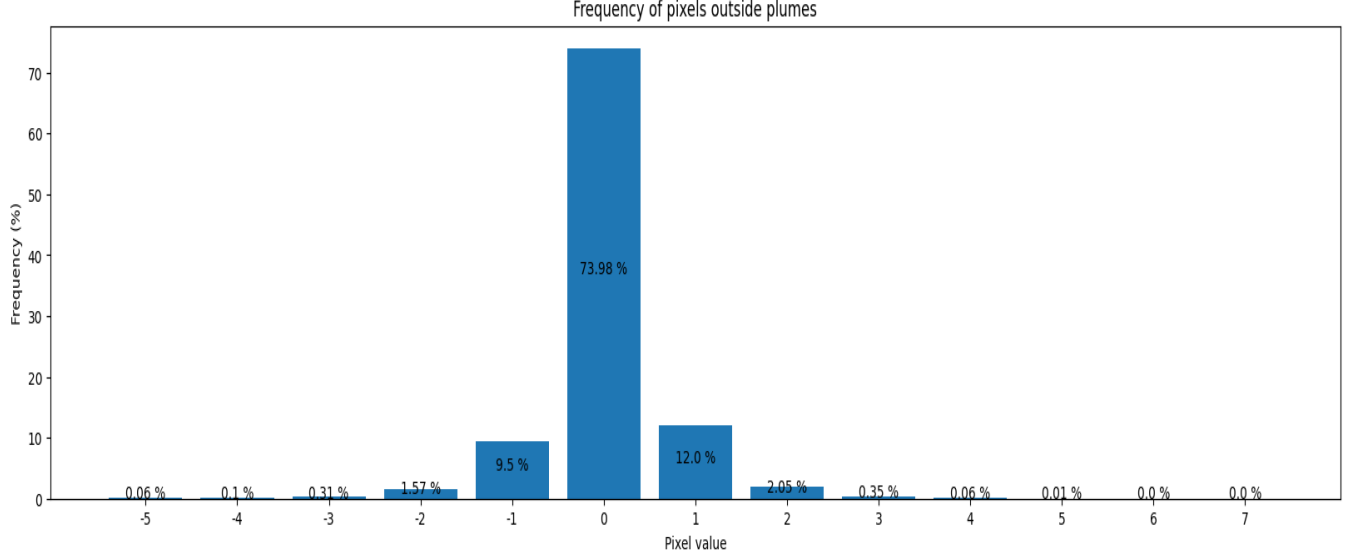


Figure 4.9 Frequencies of pixel values outside plumes for the *CH₄-SER* channel

Proportions

The average number of pixels belonging to all plumes in a retrieval is around 2600, this represents less than 1% of the pixels in the image, a very small proportion similar to that of the retrievals used in [48].

4.2.9 Correlation of channels pixel values

[16] use *Pearson correlation* to measure the correlations between the *CH₄*, *ALBEDO*, and the aerosol optical channels and use them as features for the second model.

The *Pearson correlation* coefficient (r) measures the strength and direction of the linear relationship between two variables (X , Y). The resulting value can be positive/negative indicating a positive/negative linear relationship or null indicating the non-existence of a linear relationship.

$$r = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}} \quad (4.2)$$

Pearson correlation is used to measure the correlation between the intensities presented by the 3 different channels of the dataset. To calculate the correlation between two channels, they are flattened, and each one is considered as a variable. Pearson's scores are calculated for all images in the datasets and averaged. Table 4.4 shows the results obtained.

The results show that there is an intermediate positive linear correlation between *CH₄-SER*

X	Y	r
<i>CH₄-SER</i>	<i>CH₄-C</i>	0.67
<i>CH₄-SER</i>	<i>ALBEDO</i>	0.04

Table 4.4 Average Pearson Correlation Coefficients (r) for different channels (X, Y)

and *CH₄-C*, which is logical considering that one channel is created from the other, however obtaining a score no higher shows the significant difference between the two channels (see 4.2.1).

There is no linear correlation at all between *CH₄-SER* and *ALBEDO*, the information presented by the channels is not correlated.

4.3 Channels creation

As [21, 46] highlighted the impact of noise on detection quality, 10 channels of different types are created with the aim of presenting better quality information and cleaning up the noises.

4.3.1 Intensity transformed channels

Two channels, which represent intensity level categorizations of the pixel values of the *CH₄-SER* and *CH₄-C* channels, are created. All the channels issued from the process described in this subsection are referenced as *intensity transformed channels*.

CH₄-SER-CI: By observing values of *CH₄-SER*, 5 intensity level categories $\{0, 1, 2, 3, 4\}$ are empirically defined. Such that the least intense pixels belong to the lowest category (0) and the most intense pixels belong to the highest category. All pixels belonging to the lowest interval (0) are considered *background pixels*, and pixels at other intervals are associated with different *CH₄-SER* intensity levels. This way, this channel applies a background subtraction like [46, 40] by thresholding in addition to a quantification of the *CH₄-SER* intensity levels.

CH₄-SER-CI channel is illustrated in the fourth column of figure 4.2. By transforming the *CH₄-SER* using these empirical intervals and visualizing them, we found that the plumes were easier to detect and that much of the noise was removed. In most images where there are no plumes, all pixels belong to the smallest interval corresponding to the background.

CH₄-C-CI: It's the same principle as *CH₄-SER-CI*, but for the *CH₄-C* channel. As the *CH₄-SER* and *CH₄-C* channels have different intensity values, we used different empirical intervals than those defined for *CH₄-SER-CI*. Since the *CH₄-C* channel itself is not very accurate (subsection 4.2.2), neither is the transformed channel. However, the resulting channel is less

noisy, and we noticed that in some cases this channel could be used in complement to the transformed *CH₄-SER-CI* channel.

In the resulting channels, each pixel p is an integer corresponding to one of the intensity levels defined, $p \in \{0, 1, 2, 3, 4\}$.

4.3.2 Gaussian blurred channels

Gaussian blur is applied to the *CH₄-SER* channel with different smoothing parameters: 1,2,3, to reduce the noises. Beyond 3, the resulting images were too imprecise, and it was no longer possible to visually recognize the plumes.

This blurring process results in obtaining 6 new channels. 3 *Gaussian blurred* channels *CH₄-SER-G1*, *CH₄-SER-G2* and *CH₄-SER-G3* are created from the *CH₄-SER* channel, then each of these channels is transformed in the same way as *CH₄-SER-CI* resulting in 3 *Gaussian blurred intensity transformed* channels: *CH₄-SER-G1-CI*, *CH₄-SER-G2-CI* and *CH₄-SER-G3-CI*.

4.3.3 Filtered intensity transformed channels

By visualizing *CH₄-SER-CI* and *CH₄-C-CI*, *CH₄-C-CI* seemed to be able to help filter out noisy pixels from *CH₄-SER-CI*. Indeed, for plume zones, it often presents non-background intensity pixels at the same positions as *CH₄-SER-CI*.

Based on this observation, *CH₄-C-CI* is used to filter the noisy pixels of *CH₄-SER-CI* in the resulting channel. The idea is that if a background pixel (0) is present in *CH₄-CI* and the *CH₄-SER-CI* pixel presents a smaller intensity than T ($1 \leq T \leq 4$), the resulting pixel is considered as a noise (background pixel), otherwise, *CH₄-SER-CI* pixel value is kept.

Two filtered channels, *CH₄-SER-CI-F1* and *CH₄-SER-CI-F2* are created using $T = 1$ and $T = 2$. A high value of T implies more filtered values.

4.4 HandCrafted features

Many handcrafted features have been defined based on the knowledge acquired by observing and analyzing the data to extract information from the ten channels created and from the original channels *CH₄-SER* and *CH₄-C*. Nine categories of features are proposed, and from them, seven are *image-based* meaning that they are directly extracted from the visual aspect. The two others are extracted from meta-data presented in the retrievals. This section presents them.

Table 4.5 presents each feature category, the size of its feature vector, and its identifier.

Table 4.5 Feature Categories and vector sizes

ID	Feature Category	Feature vector size	Image-based
1	Retrieval dimensions	3	Yes
2	Histogram of intense squares	420	Yes
3	Histograms of intensities	135	Yes
4	Concentration Scores	21	Yes
5	Relative statistics	300	Yes
6	Histogram of gradients	180	Yes
7	Channel correlations	12	Yes
8	Geolocation Features	4	No
9	Satellite ID	5	No
Total		1080	

4.4.1 Retrieval dimension features and satellite ID

As several image dimensions exist in the dataset, features indicating the dimension of the retrieval are extracted: the number of rows, columns, and pixels.

Additionally, the satellite ID for each retrieval is one-hot-encoded in order to exploit the possible correlations between this information and other features.

For a retrieval of N pixels, these features are extracted with a time complexity of $O(1)$.

4.4.2 Histograms of intensities

Count histograms

The count histograms of the number of pixels in each intensity level are extracted for all channels. For the not *intensity-transformed* channels, pixels are rounded to the nearest integer. Each value presented in the dataset is a histogram bin. For the *intensity-transformed* channels, each intensity level (0,1,2,3,4) is considered a bin. Count histograms are used in [39, 41] and in [16] to extract pixel count statistics as features for the second model.

For a retrieval of N pixels, these features are extracted with a time complexity of $O(N)$ for all channels.

Frequency histograms

In addition to value histograms, frequency histograms are extracted, which present the values of count histograms divided by the number of pixels in an image. Count histograms provide information about the distribution of the different intensity levels of a channel, they implicitly take into account the size of the image because a larger image will have more counts than a smaller image. However, this makes it difficult to compare the intensity distributions of images of different sizes, hence the use of frequency histograms.

A preliminary assessment showed that frequency histograms resulted in lower performance than count histograms, and did not provide an improvement when combined with count histograms. Therefore, frequency histograms are not retained as features.

Skewness

Inspired by [16], *Fisher-Pearson coefficients of Skewness* are computed in an attempt to replace each count histogram with a single coefficient. It is a measurement of the asymmetry of a distribution. Considering the average value as the center of symmetry, a distribution is asymmetric if the left and right sides are not very similar. Skew value can be negative (majority of data points concentrated on the left side), positive, or like for a normal distribution close to zero (symmetric).

A preliminary assessment showed that Skewness could not be used to replace the count histograms and, therefore these features are not retained. This is explained by the fact that this metric results in too much loss of information, considering the number of pixels contained in a single retrieval.

4.4.3 Relative statistics

In order to extract information on the relative intensity levels within a retrieval, pixels are ordered by intensity level and distributed over 20 bins according to their intensity level. Such as, the first bin contains 5% of the pixels with the highest intensities, and the last bin contains 5% of the pixels with the lowest intensities.

For each bin, statistical features are extracted: minimum, maximum, and average. This feature allows the extraction of statistical information on the relative intensity levels within a retrieval. Algorithm 3 describes the procedure.

Due to the high number of resulting features, relative statistics are only extracted for the *CH₄-SER* channel, with a time complexity $O(N\log N)$ for N pixels.

Algorithm 3 Relative statistics algorithm

```

1: Let  $P$  be the set of pixels.
2: Let  $I(p)$  be the intensity of pixel  $p$  in  $P$ .
3: Divide the pixels into 20 bins based on their intensity.
4: Each bin, denoted as  $B_i$  for  $1 \leq i \leq 20$ , is ordered by intensity.
5: The first bin ( $B_1$ ) contains 5% of the pixels with the highest intensity, and the last bin
   ( $B_{20}$ ) contains 5% of the pixels with the lowest intensity.
6: for  $i = 1$  to 20 do
7:   Calculate statistics for bin  $B_i$ :
8:    $min_i \leftarrow \min\{I(p) \mid p \in B_i\}$  ▷ Minimum intensity
9:    $max_i \leftarrow \max\{I(p) \mid p \in B_i\}$  ▷ Maximum intensity
10:   $avg_i \leftarrow \frac{1}{|B_i|} \sum_{k=1}^{|B_i|} I_k$  ▷ Average intensity
11: end for

```

4.4.4 HOG

HOG are extracted for all channels (except *ALBEDO*) as global descriptors. This means that a histogram is computed for the entire image. For each channel, a 15-bin gradient histogram is extracted. The idea is that images that do not contain plumes might exhibit lower gradient magnitudes because they contain less intensity change than plume retrievals.

This type of feature is often used as a local descriptor producing a histogram per image block, but it isn't easy to apply to the approach, since the images are not all the same size, this would produce a different number of histograms for each retrieval, resulting in feature vectors of different sizes.

4.4.5 Intensity concentration features

From observations, the plumes consist of a grouping of high-intensity pixels. For this reason, two features to measure the concentration of high-intensity pixels in small regions are defined and extracted from the *intensity-transformed* channels.

Concentrations scores

The first feature is a simple concentration score for the entire image. A weight is assigned to each non-background intensity level, the higher the intensity, the higher the weight. The idea is that each non-background pixel is a potential plume, and its emission level is quantified by summing the non-background pixels in its neighborhood. This feature is inspired by [16] which computed the integrated mass enhancement (IME) of the pixels corresponding to the plume mask according to the CNN as a feature for the second model. IME is an emission

estimation method that consists of summing the column mass enhancement of each pixel of the emission [24].

An intensity $B \in \{0, 1, 2, 3, 4\}$ is set as the background, such that all intensities lower than the chosen one are also considered background. The score of a pixel is defined as the weighted sum of non-background pixels in its neighborhood. The channel concentration score is obtained by summing the score of each non-background pixel. The concentration for N no-background intensity pixels is given as follows:

$$\text{Concentration Score} = \sum_{i=1}^N \left(\sum_{j \in V(i)} w(j) \right)$$

Where $V(i)$ are the non-background pixels in the squared neighborhood of pixel i and $w(j)$ the weight associated to the intensity of pixel j .

Three scores corresponding to different background intensities $B = \{0, 1, 2\}$ are extracted for the *intensity transformed* channels.

For a retrieval of N pixels, these features are extracted with a time complexity of $O(N)$.

Histograms of intense squares

The second feature is a histogram of intense squares, which is defined by two parameters: S the dimension of the squares, and a proportion P which symbolizes the minimum proportion of intense pixels in a square to consider it intense. The squares are generated by a sliding window across the image, such that they do not overlap. A histogram of intense squares provides the count of squares that belong to each intensity level $p \in \{0, 1, 2, 3, 4\}$.

A square of N pixels is considered to have an intensity I if :

$$\frac{\sum_i^N (p(i) \geq I)}{N} \geq P \quad (4.3)$$

Where $p(i) \in \{0, 1, 2, 3, 4\}$ is the transformed intensity level of pixel i . $p(i) \geq I$ implies that only pixels with higher or equal intensity than I are summed.

Algorithm 4 details this procedure, the algorithm generates a histogram of intense squares of parameters P and S for an intensity-transformed channel C . To create a histogram of intense squares, retrievals are padded such that they contain a finite number of squares, then the count of squares of intensity I is computed for all non-background intensity levels $I \in \{1, 2, 3, 4\}$. The parameter S controls the size of the neighborhood considered, and P controls the concentration density. Due to the nature of this operation, a square of intensity

4 is also counted as a square of intensity 3,2 and 1. If a square is not assigned to at least one intensity level in I , it is classified as a background square ($I = 0$).

Algorithm 4 Histogram of intense squares

```

1: procedure HISTOGRAMINTENSESQUARES( $S, P, C$ )
2:   Pad  $C$ , so it contains a finite number of squares of size  $S$ 
3:   Initialize an array histogram of size 5 to count squares for each intensity level.
4:   Initialize a boolean background to 1
5:   for each  $S \times S$  square window with a stride of  $N$  in the image  $C$  do
6:      $background \leftarrow 1$ 
7:     for  $I$  from 1 to 4 do ▷ Loop over non-background intensity levels
8:       Let  $A$  be the number of pixels in the square with intensity  $\geq I$ .
9:       if  $\frac{A}{N^2} \geq P$  then
10:         $histogram[I] \leftarrow histogram[I] + 1$ 
11:         $background \leftarrow 0$ 
12:      end if
13:    end for
14:    if  $background == 1$  then
15:       $histogram[0] \leftarrow histogram[0] + 1$ 
16:    end if
17:  end for
18:  Output: The histogram of intense squares
19: end procedure

```

Similarly to the approach of [41], different square sizes S are used to obtain different partitions of the retrieval, each one is responsible for detecting intensity concentrations at a certain scale. Histograms of intense squares are generated for all combinations of $P = \{0.3, 0.5, 0.8\}$ and $S = \{4, 6, 8, 10\}$, to extract this feature for different scales and different concentration tolerances.

For a retrieval of N pixels and a square dimension S , a histogram of intense squares is extracted with a time complexity of $O(N)$.

4.4.6 Channel correlations

Similarly to [16] we extracted Pearson correlation (subsection 4.2.9) between each of one of the 12 channels and the *ALBEDO*, in an attempt to exploit it. These are the only features that take into account the *ALBEDO* channel.

4.4.7 Geospatial features

Results from subsection 4.2.7 showed that the geolocation of the retrieval can provide meaningful contextual information to the model in addition to the visual information that is extracted. A plume is more likely contained in an area that frequently produces plumes.

Therefore, the clustering algorithm 2 is applied to the training set, with a distance threshold $D = 10km$, to form a database of clusters of close retrievals. It is then used to extract geolocation features for train and test sets.

A cluster is defined by 4 measures :

1. number of retrievals it contains: the information provided by a cluster with a lot of retrievals is more reliable than one from a cluster with fewer retrievals
2. total number of plumes it contains: as retrievals belonging to a cluster that contains a lot of plumes are more likely to display plumes and the opposite is also true, this measure captures this information.
3. number of retrievals with at least one plume it contains: this is a complementary measure to the previous one, but it adds more contextual information. The previous measure would not distinguish between the case where a cluster contains many retrievals that all display plumes and the case where a cluster contains many retrievals without plumes and one outlier retrieval with many plumes, but this one does.
4. position: the average position of the retrievals within it, it is used to generate the distance feature presented below.

To extract the geolocation features from a retrieval, it is attributed to the closest cluster by geodesic distance, then measures 1,2,3 (not 4) in addition to the distance between the cluster and the retrieval are used as geolocation features for the retrieval.

As a retrieval is necessarily attributed to a cluster, the distance measurement makes it possible to take their proximity into account. A very small distance means that the retrieval is located in the center of the geographic area covered by the cluster, so the information presented by the geospatial features is reliable. On the other hand, if it is very large, this means that the observation is not located in the geographical area of the cluster and that it is only the closest cluster, the information is not reliable.

It is important to note that these features are extracted for training and test observations, after which the cluster database is no longer needed.

For K retrievals, the clustering is realized with a time complexity of $O(K^2)$. For T retrievals, the feature extraction is done with a time complexity of $O(T \cdot C)$ where C is the number of clusters.

4.5 Features selection and models fine-tuning

The last step of the proposed approach consists of training models on the handcrafted features in order to distinguish the two classes of retrievals. From the perspective of this approach, classification performance can be influenced by several parameters such as the chosen subset of features, the preprocessing applied to the features, the choices of models to implement, and their hyperparameters.

Therefore, the process of selecting high-performance models was carried out in two stages. First, a preliminary phase where different subsets of features, preprocessing, and dimensionality reduction techniques are evaluated with a reduced number of models and without fine-tuning. The second phase involves training multiple models and fine-tuning their hyperparameters based on learnings from the preliminary phase.

4.5.1 Features, scaling, and dimensionality reduction assessments

In this phase, the objective is to evaluate the different designed features, data reduction methods, and the effect of feature scaling before performing further optimizations. To do this, several experiments were carried out with two arbitrary models: LR to represent the parametric methods and ADA because it is a model that is generally efficient by experience. Retrieval dimensions features (1) are always fed to the models for all experiments, the remaining features are assessed through 3 scenarios :

1. Each feature category (except 9) is tested individually to assess its individual performance. In addition, the impact of standardization (scaling) is studied in this scenario. It contains 16 test cases for each model.
2. Different combinations of image-based features are combined and tested together in order to find the best one. In addition, the impact of scaling and dimensionality reduction techniques such as PCA and SVD are studied in this scenario.
3. Geolocation features (8) are combined with image-based features except HOG (6) and tested together. In addition, the impact of adding the satellite ID feature (9) is studied.

By the end of this stage, the best feature configurations and preprocessing methods are determined and fixed to perform model selection and hyperparameter tuning.

4.5.2 Models fine-tuning

Several machine learning models are trained, fine-tuned, and tested with all the features except HOG features (6): ADA, GB, HGB, RF, bagging of DT, LR and a NN.

Features are scaled with standard scalar for parametric models. The different models are fine-tuned using Random Search with 5-fold cross-validation and 20 hyperparameters configuration tests by model. The NN is trained with the stochastic batch gradient-based optimizer ADAM until convergence using log-loss and RELU as activation functions.

Table 4.6 presents the different fine-tuned models, the search spaces, and the best hyperparameter values found on the first split. It should be noted that besides the presented ones other types of models like k-nearest neighbour, Gaussian models, or SVM were tested and were abandoned due to poor performance or 'endless' training time.

Model	Hyperparameter	Search Space	Best Values
ADA	Number of Estimators	50, 100, 150, 200, 250	200
	Learning Rate	0.01, 0.1, 0.5, 1.0	0.1
GB	Number of Estimators	50, 100, 150, 200, 250	200
	Learning Rate	0.01, 0.1, 0.5, 1.0	0.5
	Maximum Depth	5, 10, $\emptyset$	5
HGB	Number of Estimators	100, 150, 150, 200, 250	200
	Learning Rate	0.01, 0.1, 0.5, 1.0	0.1
	Maximum Depth	50, 100, $\emptyset$	$\emptyset$
RF	Number of Estimators	50, 100, 150, 200, 250	150
	Max Features by Split	sqrt, log2, $\emptyset$	$\emptyset$
	Maximum Depth	50, 100, $\emptyset$	100
	Entropy Measure	gini, log loss, $\emptyset$	log loss
Bagging of DT	Number of Estimators	50, 100, 150, 200, 250	100
	Maximum Depth	50, 100, $\emptyset$	$\emptyset$
	Entropy Measure	gini, log loss, $\emptyset$	log loss
LR	Regularization Strength	0.01, 0.01, 1, 10, 100	0.1
	Regularization Type	linear, squared	squared
NN	Number of Hidden Layers	2, ..., 10	5
	Number of Neurons by Layer	[5, 100]	50
	Batch size	50, 100, 150, 200	0.01
	Regularization Strength	0.0001, 0.001, 0.01, 0.1	0.001

Table 4.6 Hyperparameter search spaces and best configurations for ADA, GB, HGB, RF, Bagging of DT, LR and NN

Once the models are fine-tuned, they are combined into a Voting Classifier (VC) with the aim of improving performance. The use of a soft or a hard vote is tested.

4.6 Simple informative comparison: Image-based features versus CNNs

As the literature on methane plume detection primarily relies on the use of CNNs to extract visual features and classify plumes, a simple informative experiment involving comparing the performance of our image-based features to the performance of a CNN architecture is realized. The goal is to evaluate the suitability and appropriateness of the image-based features compared to the dominant approach. It is important to note that the goal is not to produce the best optimized CNN but rather to use it to provide an informative comparison between the two methods.

4.6.1 Dataset

Since the input image sizes of CNN must be uniform, a new dataset is created for this experiment. Only the retrievals having dimensions inferior to 600×600 are kept and padded to a uniform 600×600 size, resulting in a dataset of approximately 7000 retrievals, with 24% of plume retrievals, a similar proportion compared to the whole dataset (26%).

As explained in subsection 4.1.2, methods involving downsampling, dividing retrievals into patches, or using the largest dimension of the dataset as a reference point and padding the other retrievals accordingly are excluded. For this reason, the choice to define a maximum image size and to pad images of smaller size presented itself as a convenient solution. However, it implies two considerations.

First, all the presented approaches on methane plume detection [16, 48, 22, 47] used inputs smaller than 300×320 pixels, and [48] showed that the detection performance decreases rapidly as the dimensions of the inputs increase, so the maximum size should not be too large [48]. Moreover, large input requires a lot of video RAM to train deep CNN architectures. Second, choosing a size that is too small would result in a small dataset, which is also problematic according to the literature for training CNNs.

The choice of a 600×600 size showed a good balance between the size of the resulting dataset and the one of the images.

4.6.2 Experiment

The dataset is divided into training (80%) and testing (20%) sets. An HGB model that showed good results for the approach (see section 5.2) is trained only on the image-based features of the new dataset and is compared to CNNs.

For the CNNs, *ResNet* architectures are used, feature extractions rely mainly on a *ResNet* block constituted of convolution layers, skip-connections that allow deep feature extraction and batch-normalization layers that showed good results in [46, 44, 45]. After feature extraction, the feature maps are processed via average pooling and passed to fully connected layers for classification.

ResNet-18 and the deeper *ResNet-32* are implemented. *ResNet-50* is also tested but due to its depth, the training could not converge and showed very poor results. The hypothesis that explains this is that the size of the training set is too small to train a network of this depth, this was also observed in [46] experiments. As a result, deeper architectures like *ResNet-101* are not evaluated.

Training is carried out with small batches of 4 (*ResNet-18*) and 2 (*ResNet-32*) images due to the size of the inputs and the respective depths of the architectures. They are trained over 20 epochs (convergence is reached before the end) with Adam optimizer, the best parameter configurations based on the validation loss are kept and used on the test set.

The three original channels (*CH₄-SER*, *CH₄-CI*, *ALBEDO*) in addition to 4 of the created channels (*CH₄-SER-CI*, *CH₄-SER-G1*, *CH₄-SER-G2*, *CH₄-SER-G3*) are normalized like [16, 46, 7] and used as inputs.

4.7 Approach assessment

As machine learning models can present variant results depending on the distribution of the learned subset of the dataset, 5 different splits (train/test) of the original dataset are used to assess the proposed approach, [16] used a similar assessment approach.

The class balance and proposed categorization proportions (subsection 4.2.6) of the original dataset are preserved through train and test sets. The test set size is set to 20% of the 16679 retrievals for each split.

Models and feature selection are only performed on the first train/test split, once the best hyperparameter configurations are set, models are trained and assessed through each train/test split. This permits us to produce more rigorous results by taking this phenomenon into account and mitigating it as much as possible by proposing average metrics obtained on

different splits.

The evaluation metrics used are accuracy, precision, recall, f1-score, and plume retrievals detection rate for each retrieval emission rate category (subsection 4.2.6).

4.8 Detection pipeline

Once a model is trained and tested on the handcrafted features, it can be used in a detection pipeline. Figure 4.10 illustrates the prediction pipeline, constituted of the two major stages: features extraction and model prediction.

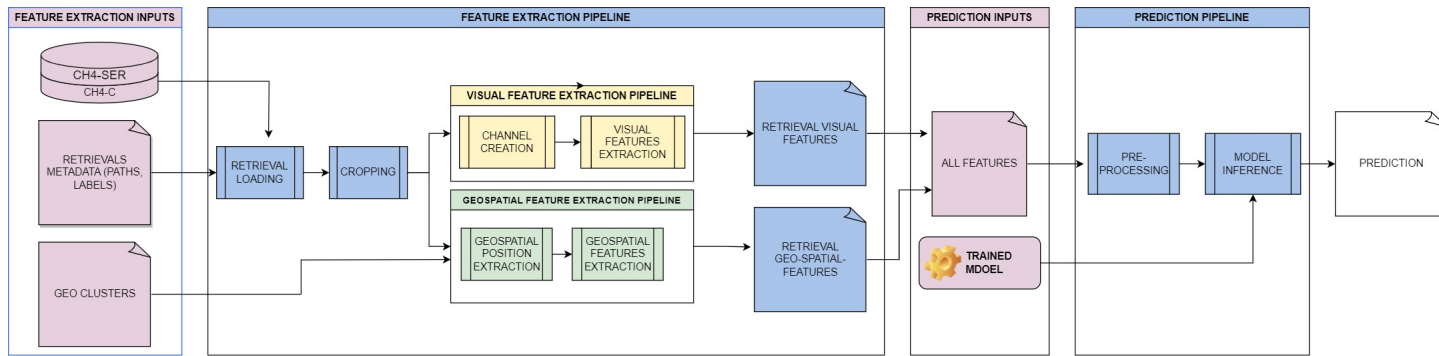


Figure 4.10 Prediction pipeline

It involves loading a retrieval, extracting handcrafted features, and using them to obtain a prediction of the retrieval class. The pipeline requires the following inputs:

- The retrieval file provided by GHGSAT, which specifies retrieval paths and satellite ID.
- The retrievals.
- A geospatial clustering database generated as explained in subsection 4.4.7.
- A pre-trained model.

The pipeline proceeds as follows:

1. Loading and cropping the *CH₄-SER* and *CH₄-C* channels from the retrievals data.
2. Concurrently, extracting both geolocation and image-based features. Image-based feature extraction involves the generation of the 10 channels.

3. These features are scaled depending on the model and are used to obtain predictions from a pre-trained model.
4. The extracted features and model predictions are saved in CSV files.

4.9 Environment and libraries

All experiments and implementations are carried out with Python 3.9. The following libraries are used to implement the solution:

- Pillow [71] is used for loading retrievals.
- NumPy [72] is used for numeric operations, array operations, data handling, and analysis.
- Scikit-learn [73] is used to implement models, standard scalar, dimensionality reduction technique, and model selection.
- Rasterio [74] is used to extract geolocation data from GeoTiff format.
- Pyproj [75] is used to project the geolocation data into the desired coordinate system.
- Geopy [76] is used to compute geodesic distance between retrievals.
- Scipy [77] is used to implement Gaussian blurring and to compute Pearson correlation score.
- Scikit-image [78] is used to compute of the gradients for the HOG features.
- Pytorch [79] is used to implement the CNNs.

CHAPTER 5 RESULTS AND DISCUSSIONS

In this chapter, the different results and their analyses are presented and discussed. First, the different features, and the effects of scaling and dimension reduction techniques are investigated. Second, the performances of the fine-tuned models are studied. Third, the results of the comparative experiment between the image-based features and the CNNs are examined. Fourth, the results of the general evaluation of the approach are discussed before finishing with a comparison to the literature.

5.1 Features, scaling, and dimensionality reduction assessments

The results of the features, scaling, and dimensionality reduction assessments are presented in this section. The objective is to evaluate the different designed features, data reduction methods, and the effect of feature scaling before performing further optimizations. To do this, only two models (LR, ADA) are trained and assessed on the first split through 3 scenarios.

It should be noted that, for the first two scenarios, the results are mainly analyzed through class 1 metrics. This is explained by the fact that the results of the various proposed experiments showed that good metrics associated with class 0 were simply achievable, but that the difficulty lies in the detection of class 1. This reduces the number of results displayed and allows a more structured analysis that relies on fewer variables.

5.1.1 Individual features analysis

This experiment consists of training and testing the two models on each feature category with and without feature scaling. The goal is to assess the pertinence of each feature alone and the impacts of SS on the results.

Tables 5.1 and 5.2 respectively report LR and ADA classification performances of class 1 for each feature category with and without feature scaling. Because the results of ADA are exactly the same with and without scaling, each row represents both experiments in table 5.2.

Feature scaling shows no impact on ADA results. For LR it seems to significantly improve performances in some cases (features 2 and 3) and does not impact performance in the other cases, it never causes a loss of performance. Therefore, the references to LR in this subsection refer to the versions with scaling.

Table 5.1 Comparison of LR classification performances for each feature with and without scaling for class 1. F-ID, Acc, Prec.1, Rec.1 F1.1, DR.1. DR.2 DR.3, refers respectively to Feature category ID, accuracy, precision for class 1, recall for class 1, f1-score for class 1, and detection rate of retrievals in categories 1,2,3.

Model	F-ID	Acc	Prec.1	Rec.1	F1.1	DR.1	DR.2	DR.3
LR	1,2	0.75	0.81	0.06	0.10	0.00	0.01	0.20
LR + SS	1,2	0.78	0.90	0.16	0.27	0.00	0.11	0.40
LR	1,3	0.83	0.86	0.41	0.55	0.09	0.37	0.80
LR + SS	1,3	0.86	0.87	0.56	0.68	0.20	0.57	0.88
LR	1,4	0.87	0.87	0.57	0.69	0.25	0.59	0.84
LR + SS	1,4	0.87	0.86	0.57	0.68	0.25	0.59	0.84
LR	1,5	0.78	0.81	0.18	0.29	0.01	0.11	0.47
LR + SS	1,5	0.78	0.85	0.18	0.30	0.01	0.12	0.46
LR	1,6	0.77	0.81	0.15	0.26	0.00	0.07	0.43
LR + SS	1,6	0.77	0.81	0.15	0.26	0.00	0.07	0.43
LR	1,7	0.74	0.12	0.00	0.00	0.00	0.00	0.00
LR + SS	1,7	0.74	0.21	0.00	0.01	0.00	0.00	0.01
LR	1,8	0.87	0.83	0.62	0.71	0.41	0.63	0.80
LR + SS	1,8	0.87	0.84	0.62	0.71	0.41	0.63	0.80

For both models, the features that perform the worst for all metrics are the channel correlations features (7). This is logical because they do not provide information on the intensities of CH4, they must be used in combination with other features to correctly exploit the information that they present in the context of our problem. These features are excluded for the remainder of this analysis.

The features that perform best overall are geospatial features (8), for both models, they present the highest f1-scores and highest recalls. Using these features, ADA shows the best results of the whole experiment in terms of accuracy (0.87), f1-score (0.75), and recall (0.83)

Table 5.2 Comparison of ADA classification performances for each feature without feature scaling for class 1

Model	F-ID	Acc	Prec.1	Rec.1	F1.1	DR.1	DR.2	DR.3
ADA	1,2	0.87	0.86	0.60	0.70	0.24	0.63	0.87
ADA	1,3	0.85	0.80	0.59	0.69	0.27	0.61	0.87
ADA	1,4	0.86	0.88	0.54	0.67	0.21	0.56	0.82
ADA	1,5	0.87	0.84	0.64	0.72	0.30	0.68	0.89
ADA	1,6	0.84	0.77	0.55	0.64	0.17	0.59	0.84
ADA	1,7	0.75	0.62	0.12	0.20	0.05	0.07	0.26
ADA	1,8	0.85	0.68	0.83	0.75	0.68	0.85	0.94

but also the worst precision (0.68) by 12% excluding features 6. Hence, these results are not really exploitable, the model emits a lot of false alarms (FP).

The best precision (90%) is obtained with LR and features (2) but it comes with a very low recall of 16%. The resulting model is ineffective because it simply labels the majority of retrievals as without plumes.

LR performs relatively poorly with features 2, 5, and 6 that present the three lowest recalls, with respectively 16%, 18%, and 15%, the common point between these three features is that they are histograms of values.

On the other, hand ADA presents respective recalls of 60 and 64%, the two highest recalls after geospatial features (8) using features 2 and 5. This means that they provide relevant information, but the LR fails to exploit it correctly. However, features 6 perform relatively low even for this model, showing the lowest f1-score and accuracy. The information they provide may be less relevant than other features in differentiating between the two classes of retrievals.

Excluding features 7 and features 2,5 and 6 for LR, accuracies vary between 0.84 and 0.87. These results mainly tell us that all features except 7 present information relevant, and that they individually permit the distinction between certain retrievals that contain plumes from those that do not. Additionally, ADA displays better f1-scores for all features (0.54 - 0.83) and better recalls except for features 4.

Finally, it is already possible to observe that the plume retrievals belonging to Category 3 are much simpler to recognize than those belonging to Category 1. Indeed, the detection rates for categories 1,2 and 3 are ascending for all features, offering up to 89% detection rates in Category 3 (excluding feature 8).

5.1.2 Combination of image-based features, scaling, and dimensionality reduction

This experiment consists of training and testing the two models on combinations of image-based features (2,3,4,5,6,7) to analyze if the different features provide complementary information and to find the combination of features that produces the best results. Furthermore, as the numerical magnitude of the different features varies a lot more compared to the previous experiment, feature scaling is still assessed.

As there are 127 combinations that can contain one or more features out of the set of 7 image-based features, the subset of retained features is determined empirically. Basically, all image-based features are retained except features 6. Indeed, the more features we add,

the more the results increase slightly except when we add or remove features 6 which do not impact the results.

As the number of features increases significantly (891 features) compared to the previous experiment, the use of two dimensionality reduction techniques is assessed: PCA and SVD. Less than 30 principal components are kept for a total 99% explained variance.

Table 5.3 reports LR and ADA classification performances with and without scaling, with and without PCA and SVD for all image-based features except features 6.

Table 5.3 Comparison of LR and ADA classification performances, with and without scaling, with and without PCA and SVD for all image-based features except features 6

Model	F-Id	Acc	Prec.1	Rec.1	F1.1	DR.1	DR.2	DR.3
LR	1,...,5,7	0.86	0.88	0.52	0.65	0.15	0.53	0.87
LR + SS	1...5,7	0.88	0.88	0.64	0.74	0.26	0.68	0.91
LR + PCA	1,...,5,7	0.70	0.43	0.60	0.50	0.35	0.57	0.88
LR + SVD	1,...,5,7	0.84	0.86	0.45	0.59	0.02	0.01	0.45
ADA	1,...,5,7	0.88	0.87	0.64	0.74	0.30	0.67	0.89
ADA + SS	1,...,5,7	0.88	0.87	0.64	0.74	0.30	0.67	0.89
ADA + PCA	1,...,5,7	0.78	0.71	0.27	0.40	0.06	0.19	0.64
ADA + SVD	1,...,5,7	0.84	0.80	0.50	0.62	0.15	0.50	0.84

PCA drastically reduces performance for both models. SVD performs better than PCA, this attests of the sparsity of the feature matrix, however, the models still perform better without dimensionality reduction.

Feature scaling has no impact on ADA, while it significantly enhances LR results. Therefore, the references to LR and to ADA in this subsection refer to the version without dimensionality reduction and with scaling for LR.

Both models perform better than using individual image-based features, demonstrating that the features provide complementary information about the data. Both models show record f1 scores of 0.74 and record accuracies of 0.88.

LR performs slightly better than ADA with a better precision by 1% and better detection rates in categories 2 and 3, detecting up to 91% of retrievals in Category 3.

From these observations, the dimensional reduction techniques and HOG features (6) are excluded and SS will be applied to parametric models for the rest of the work.

5.1.3 All features combined

This experiment consists of training and testing the two models on all feature categories except 6 and with and without satellite ID (9).

Tables 5.4 and 5.5 reports respectively LR and ADA classification results and detection rates for image-based features only, all features except satellite ID and all features.

Table 5.4 Comparison of LR and ADA classification performances for image-based features only, all features with and without satellite ID

Model	F-Id	Acc	Prec.1	Rec.1	F1.1	Prec.0	Rec.0	F1.0
LR + SS	1...,5,7	0.88	0.88	0.64	0.74	0.88	0.97	0.93
LR + SS	1...,5,7,8	0.91	0.86	0.76	0.81	0.92	0.96	0.94
LR + SS	1...,5,7,8,9	0.91	0.86	0.76	0.81	0.92	0.96	0.94
ADA	1...,5,7	0.88	0.87	0.64	0.74	0.88	0.97	0.92
ADA	1...,5,7,8	0.91	0.83	0.81	0.82	0.94	0.94	0.94
ADA	1...,5,7,8,9	0.91	0.83	0.81	0.82	0.94	0.94	0.94

Table 5.5 Comparison of LR and ADA detection rates performances for image-based features only and all features

Model	F-Id	DR.1	DR.2	DR.3
LR + SS	1...,5,7	0.26	0.68	0.91
LR + SS	1...,5,7,8	0.47	0.79	0.97
ADA	1...,5,7	0.30	0.67	0.89
ADA	1...,5,7,8	0.58	0.86	0.95

The results show that combining all the features provides a record accuracy of 91% for both models and records f1-scores compared to the previous experiments, attesting to the balance of the resulting models. Both models manage to detect almost all retrievals belonging to category 3. The addition of satellite IDs in the feature matrix does not impact the results of the two models at all.

For LR, using image-based features only gives f1-scores and recall for class 1 of 0.74 and 0.64 which scales to 0.81 and 0.76 when incorporating geospatial features. Similarly, for ADA these metrics, go from 0.74 and 0.64 to 0.82 and 0.81.

Regarding class 0, both methods present the same high f1-score of 0.94. The difference between the two models is that LR produces more false alarms on class 0 while ADA detects fewer retrievals belonging to this class, this illustrates the tradeoff between the precision and the recall.

For class 1, ADA presents a slightly better f1-score (0.82) by 1%. Again, the difference between the two models resides mainly in the precision and recall tradeoff, the difference is that ADA produces more false alarms than (LR) which detects fewer plume retrievals. However, while ADA detects significantly more plumes in both categories 1 and 2, LR detects more plumes in Category 3.

The combination of all features gives good classification performance for class 1 and very good performance for class 0 and significantly increases performance compared to using image-based features or geospatial features alone. The different features proposed express relevant information and allow the distinction between plume and no-plume retrievals without the need for hyperparameters optimization.

5.2 Models fine-tuning

This experiment consists of training and fine-tuning several machine learning models with all the features except HOG features (6): ADA, GB, HGB, RF, bagging of DT, LR and a NN. Features are scaled with SS for parametric models and the different models are fine-tuned using Random Search with 5-fold cross-validation and 20 hyperparameters configuration tests by model.

First, similarly to [16], it was observed that the performance does not vary much from one hyperparameter configuration to another for all models. Second, balancing retrieval classes by assigning more weight to class 1 reduces the performances for most models without improving the performance of any model.

Tables 5.6 and 5.7 present the classification performances and detection rates for the best hyperparameter values presented in table 4.6 and for the VCs.

Table 5.6 Comparison of the fine-tuned models classification performances

Model	Acc	Prec.1	Rec.1	F1.1	Prec.0	Rec.0	F1.0
ADA	0.91	0.84	0.79	0.82	0.93	0.95	0.94
GB	0.91	0.85	0.82	0.83	0.94	0.95	0.94
HGB	0.92	0.87	0.83	0.85	0.94	0.96	0.95
RF	0.92	0.86	0.81	0.84	0.94	0.95	0.94
BAGGING DT	0.89	0.93	0.65	0.76	0.89	0.98	0.93
LR + SS	0.91	0.87	0.75	0.81	0.92	0.96	0.94
NN + SS	0.92	0.84	0.84	0.84	0.94	0.94	0.94
VC + SOFT	0.92	0.89	0.80	0.84	0.93	0.97	0.95
VC + HARD	0.92	0.88	0.81	0.84	0.94	0.96	0.95

Table 5.7 Comparison of the fine-tuned models detection rates

Model	DR.1	DR.2	DR.3
ADA	0.56	0.84	0.95
GB	0.59	0.86	0.96
HGB	0.60	0.87	0.97
RF	0.57	0.87	0.95
BAGGING DT	0.30	0.69	0.90
LR + SS	0.45	0.80	0.95
NN + SS	0.61	0.88	0.97
VC + SOFT	0.53	0.85	0.96
VC + HARD	0.53	0.85	0.96

All models display high accuracies, with the lowest score of 89% and the highest score of 92% held by several models. This means that the overall predictions are accurate and will misclassify less than 1 out of 10 predictions.

The bagging classifier performs worst on all metrics for class 1 except for precision, for which it achieves the best score. However, it has the worst f1-score and recall by respectively 5 and 10% and displays the lowest accuracy. This means that its high precision comes from the fact that it does not produce much class 1 prediction. For this reason, bagging of DT is no longer considered for the rest of the discussion.

LR performs weakly relative to the other classifiers for class 1, it shows the worst recall of 75% by 4% and displays the lowest f1-score by 4% and does not display particularly high results for class 0. LR is also excluded for the rest of the discussion

For class 0, f1-scores, precisions and recalls vary respectively between: 0.94 and 0.95, 0.94 and 0.97, and 0.92 and 0.94. The models produce very similar results for this class, some are more precise while others offer higher detection rates.

On the other hand, the f1-scores, precisions and recalls of class 1 vary respectively between: 0.82 and 0.85, 0.84 and 0.89, and 0.84 and 0.87. This shows that for this class, performance oscillates more from one model to another compared to class 0.

The soft and hard VCs possess near identical performances and display the best accuracy, the best precisions for class 1 (89% and 88%) the best f1-score for class 0 but show relatively low recalls (80% and 81%) for class 1 compared to other models.

HGB stands out from other models by offering the highest accuracy, the highest f1-scores for both classes, the best detection rate for Category 3, and the most precision for class 1 excluding VCs. RF and GB produce for both classes slightly lower but very similar results

to HGB. RF performs generally better than GB and shows a better accuracy by 1%. ADA performs similarly but less well than GB.

Similarly, the NN shows performances that stand out with the best accuracy, the best recall for class 1, the best precision for class 0, and the best detection rates in the 3 categories while displaying f1-scores less than 1% of the maximums for both classes.

5.3 Simple informative comparison: Image-based features versus CNNs

This experiment consists of comparing the performances of our image-based features to those of CNNs. *ResNet-18*, *ResNet-32*, and a HGB are trained and tested on a new dataset of approximately 7000 600×600 retrievals.

It must be emphasized that within the framework of this experiment, the quest for performance with CNNs was not pushed to its maximum, some preliminary experiments were used to define the right configurations before obtaining the presented models. For this reason, the presented CNN's performances may be higher by increasing the size of the training set with data augmentation like some presented related works and by implementing more complex and deep architectures.

For the CNNs, during preliminary experiments, the use of *CH4-C* and *ALBEDO* channels always reduced performances, using the remaining 4 channels showed the best results. Table 5.8 presents the classification performances and detection rates for the implemented ResNet architectures and the HGB trained only on the image-based features.

Table 5.8 Comparison of the classification performances and detection rates of ResNet architectures and HGB on image-based features

Model	Acc	Prec.1	Rec.1	F1.1	Prec.0	Rec.0	F1.0	DR.1	DR.2	DR.3
RN-18	0.89	0.87	0.64	0.74	0.89	0.97	0.93	0.17	0.69	0.90
RN-32	0.90	0.85	0.71	0.77	0.91	0.96	0.94	0.30	0.81	0.86
HGB	0.89	0.84	0.69	0.76	0.91	0.96	0.93	0.40	0.72	0.89

In terms of accuracies, the three methods present very similar results in the advantage for ResNet-32 (0.90) by 1%. The behaviors for class 0 are also very similar, f1 scores vary by 1% in the advantage, again for ResNet-32 (0.94).

For class 1, the results vary more. ResNet-18 shows the best precision but the lowest recall (0.64) by 5% and 7% compared respectively to HGB and ResNet-32. ResNet-32 shows the best f1 score of 0.77, 1, and 3% more compared to HGB and ResNet-18.

Concerning detection rates, HGB detects more retrievals belonging to Category 1 (40%)

than the other methods. For this category, ResNet-18 presents a very low detection rate of 17% and ResNet-30 presents a detection rate of 30%, 10% less than HGB. For Category 3, ResNet-18 presents the best result (90%) followed by HGB (89%) while ResNet-32 performs slightly lower (86%). For Category 2, ResNet-32 performs the best by 9% (81%), followed by HGB that performs 3% better than ResNet-18.

This experiment shows that our image-based features present quite similar results to CNN architectures, proving that they provide meaningful visual information. While ResNet-32 is overall the best approach, the image-based features present a better detection rate by 10% for retrievals in Category 1. One hypothesis to explain this is that some plumes appearing in this category are so small that the CNNs fail to describe their shape, while the high-intensity pixels associated with them are always captured by the features 3.

Furthermore, this experiment also shows, that due to the presence of numerous visual phenomena such as artifacts or very small plumes (see subsection 4.2.2) detecting the presence or absence of plumes based solely on visual characteristics has these limitations. Hence, the use of Geospatial features (8) and their important contribution.

5.4 Approach assessment

To evaluate the proposed approach more rigorously, the fine-tuned models presented in section 5.2 are assessed through different train/test splits data distributions. The models are only fine-tuned on the first split and are trained and assessed through the average performances of 5 different splits, including the first one.

For this reason, the results presented in this section are more reliable than those presented in section 5.2, and are considered the definitive results of the proposed approach. Tables 5.9 and 5.10 present the average classification performances and detection rates for the fine-tuned models over the 5 splits.

The first observation is that the average results are slightly lower in general than those present for the first split (tables 5.6, 5.7). Indeed, it shows generally lower performance in terms of accuracy, recall, and f1-score for class 1 and in terms of detection rates. However, the results are nearly identical for class 0. This is illustrated by a general drop of 1% in the accuracies of most models.

This is mainly explained by the difference in the data distribution contained in each split. Some splits may have harder test sets than others, containing many complex retrievals to label, or maybe simply present training sets that do not capture all the data distributions present in the test set. This is exactly why a similar experiment is carried out in [16].

Table 5.9 Comparison of the fine-tuned models classification performances averaged over 5 splits

Model	Acc	Prec.1	Rec.1	F1.1	Prec.0	Rec.0	F1.0
ADA	0.90	0.84	0.78	0.81	0.93	0.95	0.94
GB	0.90	0.83	0.80	0.82	0.93	0.94	0.94
HGB	0.91	0.85	0.80	0.83	0.93	0.95	0.94
RF	0.91	0.86	0.80	0.82	0.93	0.95	0.94
BAGGING DT	0.89	0.92	0.63	0.75	0.89	0.98	0.93
LR + SS	0.90	0.88	0.72	0.79	0.91	0.97	0.94
NN + SS	0.91	0.87	0.76	0.81	0.92	0.96	0.94
VC + SOFT	0.92	0.89	0.78	0.83	0.92	0.97	0.95
VC + HARD	0.92	0.89	0.78	0.83	0.93	0.97	0.95

Table 5.10 Comparison of the fine-tuned models detection rates averaged over 5 splits

Model	DR.1	DR.2	DR.3
ADA	0.52	0.83	0.96
GB	0.55	0.85	0.94
HGB	0.56	0.84	0.96
RF	0.56	0.84	0.95
BAGGING DT	0.23	0.60	0.73
LR + SS	0.42	0.75	0.94
NN + SS	0.50	0.80	0.95
VC + SOFT	0.50	0.82	0.96
VC + HARD	0.52	0.83	0.96

Excluding NN, except for a slight overall drop in performance for class 1, the conclusions drawn regarding the relative results of the different models are similar to those presented in the section 5.2.

Indeed, while NN performs best in split 1 in terms of detection rates and recall for class 1, it performs less well on these metrics across the different splits compared to other models, except for the detection rate of the Category 3 where it still presents the best results. This is the model that experienced the greatest changes in performance between the first split and multiple-split evaluations. This may be explained by the fact that the model was heavily fine-tuned on validation distributions specific to the first train split. However, this evaluation shows that the model still remains efficient with an accuracy of 0.91 and f1-scores of 0.81 and 0.94 for classes 1 and 0.

From this evaluation, the models that perform best are HGB and the VCs. VCs present the best accuracies (0.92), 1% more than that of HGB. For class, 1, the 3 models present the

best f1-scores of 0.83, HGB produces the best recall (0.80) which is 2% higher than those of VCs, but VCs present the best precision for this class (excluding bagging of DT). VCs have the best f1-scores (0.95) for class 0 which are 1% higher than that of HGB, which explains the difference between their accuracies. Although the results of the two VCs are extremely similar, hard voting performs a little better than soft voting (DR.1, Prec.0).

Ultimately, the difference between VCs and HGB is very small. VCs are more precise while HGB has a higher detection rate for plume retrievals, but it emits more false alarms. The choice of model can be adapted to industrial priority.

This assessment proves that the proposed approach is robust through different train/test data distributions. It produces accurate distinctions between plume and no-plume retrievals. Less than 1 out of 10 retrievals are misclassified, 80% of plume retrievals are detected and 96% of the 25% of plume retrievals containing the largest emission rates are detected. If the plume retrievals belonging to Category 1, those that are usually hard to detect confidently even for humans (see row 1 of figure 4.6), are excluded, an average plume retrievals detection rate of 88% is obtained.

5.5 Comparison with literature

First, through the different experiments, we can observe that similarly to [21, 46, 7, 16], the magnitude of the emission rate is an important factor in plume detection. Indeed, for all experiments and all models, the detection rates are increasing with respect to the emission rate category to which the retrieval belongs. The best models are capable of detecting 96 and 83% of plume retrievals belonging to categories 2 and 3 but only 52% of those contained in Category 1.

[46] and [7] respectively propose detection rates of 93% and 90%. [47] shows a detection rate of 42%. [48] presents a f1-score of 0.92 for class 1. [16] displays 0.91 in recall and 0.93 in precision on a test augmented dataset for class 1 but only a precision between 0.61 and 0.76 on a different non-augmented dataset.

So, with a plume detection rate of 80%, the best model HGB (based on recall of class 1) performs lower than all methods except [47] and [21] depending on whether we consider the results on the augmented dataset or not. However, this comparison must be taken under several considerations.

First, [46, 7] results are obtained on CH₄ video leaks using an infrared camera where leaks generally take up a large proportion of the image and are from a very short distance from the lens. This is not the case for our data. Second, the video leaks are extracted from

an experimental setup that does not take into account a wide diversity of environmental fluctuations and excludes elements that could disturb the quality of the frames. This removes numerous factors that can influence the quality of the plumes and their detection. Third, the temporal relationship between video frames is exploited to achieve better performances.

Second, [21] results are obtained on 300×300 retrievals covering $1.5 \times 1.5 \text{ km}^2$ while ours are obtained on larger retrievals that cover a wider spatial region for an average of $12 \times 12 \text{ km}^2$. This means that a plume leak will appear more accurately on [21] retrievals and will cover a greater proportion of the image compared to ours. This is supported by the fact that [21] is capable of detecting 50% of the plume with emission rates around 50 kg/h while this source rate is considered the detection limit (smallest plume detectable) of GHGSAT measuring instrument [80].

Similar considerations can be drawn for [48] results, in addition, RGB channels or hyperspectral bands are used, allowing the use of more raw information to perform the task but also requiring a lot of computing power in the case of the spectral band. Likewise, [16] uses additional aerosol optical thickness and wind data.

Third, geolocation data are used in our approach, which is not the case for any of the compared methods.

Fourth, all presented methods use fixed and much smaller input sizes than ours, and [48] showed that a smaller input size leads to better detection.

Finally, it is important to remember that all the related works treat a **different task** than ours, they focus on the classification or segmentation of individual plumes while our method classifies whole retrievals that may contain several plumes and can work with different input sizes. Consequently, the comparison has its limits because it opposes different tasks. For example, the retrievals detected by HGB as including plumes contain around 90% of the plumes in the test set, but it is incorrect to say that we can detect 90% of the plumes.

5.6 Outlier retrievals detection

The *likelihood of computational profiles* was employed alongside the neural network developed using handcrafted features to identify outlier or erroneous retrievals. A second dataset, comprising nearly 10,000 flawed retrievals labeled by GHGSATs operators, was utilized to gauge the performance of outlier detection. These retrievals may be faulty due to two primary reasons: adverse weather impacting retrieval quality or errors in acquisition sensors.

Using the neural network in conjunction with the *likelihood of computational profiles*, 88% of these retrievals were identified as being out of distribution. This substantiates the method's

effectiveness in filtering out a significant portion of flawed observations.

CHAPTER 6 CONCLUSION

To distinguish between CH₄ satellite images that contain plumes from those that don't, an approach that works with inputs of varying sizes based on the extraction of handcrafted features with traditional machine-learning models was designed from dataset analysis and observations.

Features are extracted from original and generated channels and from the geospatial positions of the retrievals. Then, they are used to train and fine-tune multiple machine-learning models. The evaluation of the approach across different distributions of the dataset shows that it produces an accurate distinction of 92% between plume and no-plume retrievals. Less than 1 out of 10 retrievals is misclassified, 80% of plume retrievals are detected, and 96% of the 25% plume retrievals containing the largest emission rates are detected. If the plume retrievals belonging to the low emission category, those that are usually hard to detect confidently even for humans are excluded, an average detection rate of 88% is obtained.

6.1 Limitations

The proposed approach presents some limitations. First, it does not take into account the textures and shapes of the plumes. Hence, it cannot distinguish from visual features an artifact that contains a lot of high-intensity pixels but which does not have the shape of a plume from a plume.

Second, it exploits geospatial features (section 5.1), which require a large database of nearby retrieval clusters to work properly. This is not easily achievable, given that the lack of labeled data is a recurring problem for hyperspectral captured data [52].

Thirdly, it does not allow the detection of individual plumes but of retrievals containing plumes, this means that if a retrieval containing several plumes is not detected by the method, many plumes would be missed at once. However, this is a consequence of the scope of the work and not of the approach itself, which could be used on image patches containing single plumes.

6.2 Future work

There are several avenues for future work. First, to improve the approach, shape descriptors could be exploited, which could help distinguish plumes from artifacts. For example, it

would be interesting to automatically extract shape features with a CNN and use them in combination with the designed features. In addition, our approach could be evaluated with data from other hyperspectral retrievals (AVIRIS, PRISMA, TROPOMI, etc. . .) or to detect emissions of other gases such as carbon dioxide for example.

Second, given the relevance of geolocation features, launching a global initiative to establish a comprehensive database listing the locations of identified emissions appears to be an attractive prospect. Such a resource could serve various purposes in various applications.

Third, exploring the potential of leveraging the *likelihood of computational profile* to improve the reliability and robustness of machine learning models presents an interesting avenue. This could include automating the detection of erroneous/outlier retrievals, identifying corrupted images such as those from adversarial attacks, or even providing prediction confidence levels that could be factored into the GHGSAT industrial process. For example, unreliable predictions could be manually inspected by operators to confirm the model diagnosis.

Finally, CNN methods could be used on retrieval patches to classify individual plumes, the performance of our approach could also be evaluated with image patches and compared. In addition, individual plume segmentation methods such as *Mask-RCNN* or *U-Net* can be implemented to directly locate and identify plumes in retrievals.

REFERENCES

- [1] I. P. on Climate Change (IPCC), *Summary for Policymakers : Global Warming of 1.5°C*. Cambridge University Press, 2022, p. 1–24.
- [2] M. Saunio, P. Bousquet, B. Poulter, A. Peregon, P. Ciais, J. G. Canadell, E. J. Dlugokencky, G. Etiope, D. Bastviken, S. Houweling, G. Janssens-Maenhout, F. N. Tubiello, S. Castaldi, R. B. Jackson, M. Alexe, V. K. Arora, D. J. Beerling, P. Bergamaschi, D. R. Blake, G. Brailsford, V. Brovkin, L. Bruhwiler, C. Crevoisier, P. Crill, K. Covey, C. Curry, C. Frankenberg, N. Gedney, L. Höglund-Isaksson, M. Ishizawa, A. Ito, F. Joos, H.-S. Kim, T. Kleinen, P. Krummel, J.-F. Lamarque, R. Langenfelds, R. Locatelli, T. Machida, S. Maksyutov, K. C. McDonald, J. Marshall, J. R. Melton, I. Morino, V. Naik, S. O'Doherty, F.-J. W. Parmentier, P. K. Patra, C. Peng, S. Peng, G. P. Peters, I. Pison, C. Prigent, R. Prinn, M. Ramonet, W. J. Riley, M. Saito, M. Santini, R. Schroeder, I. J. Simpson, R. Spahni, P. Steele, A. Takizawa, B. F. Thornton, H. Tian, Y. Tohjima, N. Viovy, A. Voulgarakis, M. van Weele, G. R. van der Werf, R. Weiss, C. Wiedinmyer, D. J. Wilton, A. Wiltshire, D. Worthy, D. Wunch, X. Xu, Y. Yoshida, B. Zhang, Z. Zhang, and Q. Zhu, “The global methane budget 2000–2012,” *Earth System Science Data*, vol. 8, no. 2, pp. 697–751, Dec. 2016. [Online]. Available: <https://doi.org/10.5194/essd-8-697-2016>
- [3] D. T. Shindell, J. S. Fuglestedt, and W. J. Collins, “The social cost of methane: theory and applications,” *Faraday Discussions*, vol. 200, pp. 429–451, 2017. [Online]. Available: <https://doi.org/10.1039/c7fd00009j>
- [4] T. Stocker, D. Qin, G.-K. Plattner, M. Tignor, S. Allen, J. Boschung, A. Nauels, Y. Xia, V. Bex, and P. Midgley, “Climate change 2013: The physical science basis,” in *Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge, United Kingdom and New York, NY, USA: Cambridge University Press, 2013, p. 1535.
- [5] “Paris agreement,” United Nations Framework Convention on Climate Change, 2015. [Online]. Available: <https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement>
- [6] R. A. Alvarez, D. Zavala-Araiza, D. R. Lyon, D. T. Allen, Z. R. Barkley, A. R. Brandt, K. J. Davis, S. C. Herndon, D. J. Jacob, A. Karion, E. A. Kort, B. K. Lamb, T. Lauvaux,

- J. D. Maasakkers, A. J. Marchese, M. Omara, S. W. Pacala, J. Peischl, A. L. Robinson, P. B. Shepson, C. Sweeney, A. Townsend-Small, S. C. Wofsy, and S. P. Hamburg, "Assessment of methane emissions from the u.s. oil and gas supply chain," *Science*, vol. 361, no. 6398, pp. 186–188, 2018.
- [7] J. Wang, J. Ji, A. P. Ravikumar, S. Savarese, and A. R. Brandt, "VideoGasNet: Deep learning for natural gas methane leak classification using an infrared camera," *Energy*, vol. 238, p. 121516, Jan. 2022. [Online]. Available: <https://doi.org/10.1016/j.energy.2021.121516>
- [8] D. Zavala-Araiza, D. R. Lyon, R. A. Alvarez, K. J. Davis, R. Harriss, S. C. Herndon, A. Karion, E. A. Kort, B. K. Lamb, X. Lan, A. J. Marchese, S. W. Pacala, A. L. Robinson, P. B. Shepson, C. Sweeney, R. Talbot, A. Townsend-Small, T. I. Yacovitch, D. J. Zimmerle, and S. P. Hamburg, "Reconciling divergent estimates of oil and gas methane emissions," *Proceedings of the National Academy of Sciences*, vol. 112, no. 51, pp. 15 597–15 602, Dec. 2015. [Online]. Available: <https://doi.org/10.1073/pnas.1522126112>
- [9] A. R. Brandt, G. A. Heath, E. A. Kort, F. O'Sullivan, G. Petron, S. M. Jordaan, P. Tans, J. Wilcox, A. M. Gopstein, D. Arent, S. Wofsy, N. J. Brown, R. Bradley, G. D. Stucky, D. Eardley, and R. Harriss, "Methane leaks from north american natural gas systems," *Science*, vol. 343, no. 6172, pp. 733–735, Feb. 2014. [Online]. Available: <https://doi.org/10.1126/science.1247045>
- [10] Y.-K. Hsu, T. VanCuren, S. Park, C. Jakober, J. Herner, M. FitzGibbon, D. R. Blake, and D. D. Parrish, "Methane emissions inventory verification in southern california," *Atmospheric Environment*, vol. 44, no. 1, pp. 1–7, Jan. 2010. [Online]. Available: <https://doi.org/10.1016/j.atmosenv.2009.10.002>
- [11] "GOSAT," <https://www.gosat.nies.go.jp/>, 2023.
- [12] A. J. Turner, D. J. Jacob, K. J. Wecht, J. D. Maasakkers, E. Lundgren, A. E. Andrews, S. C. Biraud, H. Boesch, K. W. Bowman, N. M. Deutscher, M. K. Dubey, D. W. T. Griffith, F. Hase, A. Kuze, J. Notholt, H. Ohyama, R. Parker, V. H. Payne, R. Sussmann, C. Sweeney, V. A. Velazco, T. Warneke, P. O. Wennberg, and D. Wunch, "Estimating global and north american methane emissions with high spatial resolution using GOSAT satellite data," *Atmospheric Chemistry and Physics*, vol. 15, no. 12, pp. 7049–7069, Jun. 2015. [Online]. Available: <https://doi.org/10.5194/acp-15-7049-2015>

- [13] J. A. de Gouw, J. P. Veefkind, E. Roosenbrand, B. Dix, J. C. Lin, J. Landgraf, and P. F. Levelt, “Daily satellite observations of methane from oil and gas production regions in the united states,” *Scientific Reports*, vol. 10, no. 1, Jan. 2020. [Online]. Available: <https://doi.org/10.1038/s41598-020-57678-4>
- [14] O. Schneising, M. Buchwitz, M. Reuter, S. Vanselow, H. Bovensmann, and J. P. Burrows, “Remote sensing of methane leakage from natural gas and petroleum systems revisited,” *Atmospheric Chemistry and Physics*, vol. 20, no. 15, pp. 9169–9182, Aug. 2020. [Online]. Available: <https://doi.org/10.5194/acp-20-9169-2020>
- [15] J. D. Maasakkers, D. J. Varon, A. Elfarsdóttir, J. McKeever, D. Jervis, G. Mahapatra, S. Pandey, A. Lorente, T. Borsdorff, L. R. Foorthuis, B. J. Schuit, P. Tol, T. A. van Kempen, R. van Hees, and I. Aben, “Using satellites to uncover large methane emissions from landfills,” *Science Advances*, vol. 8, no. 32, Aug. 2022. [Online]. Available: <https://doi.org/10.1126/sciadv.abn9683>
- [16] B. J. Schuit, J. D. Maasakkers, P. Bijl, G. Mahapatra, A.-W. van den Berg, S. Pandey, A. Lorente, T. Borsdorff, S. Houweling, D. J. Varon, J. McKeever, D. Jervis, M. Girard, I. Irakulis-Loitxate, J. Gorroño, L. Guanter, D. H. Cusworth, and I. Aben, “Automated detection and monitoring of methane super-emitters using satellite data,” *Atmospheric Chemistry and Physics*, vol. 23, no. 16, pp. 9071–9098, 2023. [Online]. Available: <https://acp.copernicus.org/articles/23/9071/2023/>
- [17] (2023) GHGSat. [Online]. Available: <https://www.ghgsat.com/en/who-we-are/>
- [18] (2023) GHG Satellites. [Online]. Available: <https://earth.esa.int/eogateway/missions/ghgsat/description>
- [19] P. Joyce, C. R. Villena, Y. Huang, A. Webb, M. Gloor, F. H. Wagner, M. P. Chipperfield, R. B. Guilló, C. Wilson, and H. Boesch, “Using a deep neural network to detect methane point sources and quantify emissions from PRISMA hyperspectral satellite images,” Nov. 2022. [Online]. Available: <https://doi.org/10.5194/egusphere-2022-924>
- [20] (2023) AVIRIS. [Online]. Available: <https://aviris.jpl.nasa.gov/index.html>
- [21] S. Jongaramrungruang, A. K. Thorpe, G. Matheou, and C. Frankenberg, “MethaNet – an AI-driven approach to quantifying methane point-source emission from high-resolution 2-d plume imagery,” *Remote Sensing of Environment*, vol. 269, p. 112809, Feb. 2022. [Online]. Available: <https://doi.org/10.1016/j.rse.2021.112809>

- [22] S. Kumar, C. Torres, O. Ulutan, A. Ayasse, D. Roberts, and B. Manjunath, “Deep remote sensing methods for methane detection in overhead hyperspectral imagery,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1765–1774.
- [23] S. Jongaramrungruang, C. Frankenberg, G. Matheou, A. K. Thorpe, D. R. Thompson, L. Kuai, and R. M. Duren, “Towards accurate methane point-source quantification from high-resolution 2-d plume imagery,” *Atmospheric Measurement Techniques*, vol. 12, no. 12, pp. 6667–6681, Dec. 2019. [Online]. Available: <https://doi.org/10.5194/amt-12-6667-2019>
- [24] D. J. Varon, D. J. Jacob, J. McKeever, D. Jervis, B. O. A. Durak, Y. Xia, and Y. Huang, “Quantifying methane point sources from fine-scale satellite observations of atmospheric methane plumes,” *Atmospheric Measurement Techniques*, vol. 11, no. 10, pp. 5673–5686, Oct. 2018. [Online]. Available: <https://doi.org/10.5194/amt-11-5673-2018>
- [25] D. J. Varon, D. J. Jacob, D. Jervis, and J. McKeever, “Quantifying time-averaged methane emissions from individual coal mine vents with GHGSat-d satellite observations,” *Environmental Science & Technology*, vol. 54, no. 16, pp. 10 246–10 253, Jul. 2020. [Online]. Available: <https://doi.org/10.1021/acs.est.0c01213>
- [26] D. J. Varon, D. Jervis, J. McKeever, I. Spence, D. Gains, and D. J. Jacob, “High-frequency monitoring of anomalous methane point sources with multispectral sentinel-2 satellite observations,” *Atmospheric Measurement Techniques*, vol. 14, no. 4, pp. 2771–2785, 2021. [Online]. Available: <https://amt.copernicus.org/articles/14/2771/2021/>
- [27] E. Merlo, M. Marhaba, F. Khomh, H. B. Braiek, and G. Antoniol, “Models of computational profiles to study the likelihood of dnn metamorphic test cases,” 2021. [Online]. Available: <https://arxiv.org/abs/2107.13491>
- [28] M. Marhaba, E. Merlo, F. Khomh, and G. Antoniol, “Identification of out-of-distribution cases of cnn using class-based surprise adequacy,” in *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*, ser. CAIN ’22. New York, NY, USA: Association for Computing Machinery, 2022, p. 39–40. [Online]. Available: <https://doi.org/10.1145/3522664.3528617>
- [29] C. M., *Pattern Recognition and Machine Learning*, 1st ed., ser. Information Science and Statistics. New York, NY: Springer, 2006.

- [30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, p. 1929–1958, jan 2014.
- [31] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML’15. JMLR.org, 2015, p. 448–456.
- [32] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996. [Online]. Available: <https://doi.org/10.1023/a:1018054314350>
- [33] —, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: <https://doi.org/10.1023/a:1010933404324>
- [34] Y. Freund and R. E. Schapire, “A short introduction to boosting,” 1999. [Online]. Available: <https://api.semanticscholar.org/CorpusID:9621074>
- [35] A. Natekin and A. Knoll, “Gradient boosting machines, a tutorial,” *Frontiers in Neurorobotics*, vol. 7, 2013. [Online]. Available: <https://doi.org/10.3389/fnbot.2013.00021>
- [36] J. Gao, Y. Yang, P. Lin, and D. S. Park, “Computer vision in healthcare applications,” *Journal of Healthcare Engineering*, vol. 2018, pp. 1–4, 2018. [Online]. Available: <https://doi.org/10.1155/2018/5157020>
- [37] A. Tolba, A. El-Baz, and A. El-Harby, “Face recognition: A literature review,” *International Journal of Signal Processing*, vol. 2, pp. 88–103, 01 2005.
- [38] E. P. Ijjina, D. Chand, S. Gupta, and G. K, “Computer vision-based accident detection in traffic surveillance,” 2019. [Online]. Available: <https://arxiv.org/abs/1911.10037>
- [39] T. Çelik, H. Özkaramanlı, and H. Demirel, “Fire and smoke detection without sensors: Image processing based approach,” in *2007 15th European Signal Processing Conference*, 2007, pp. 1794–1798.
- [40] P. Barmpoutis, K. Dimitropoulos, and N. Grammalidis, “Smoke detection using spatio-temporal analysis, motion modeling and dynamic texture recognition,” in *2014 22nd European Signal Processing Conference (EUSIPCO)*, 2014, pp. 1078–1082.

- [41] F. Yuan, “A double mapping framework for extraction of shape-invariant features based on multi-scale partitions with adaboost for video smoke detection,” *Pattern Recognition*, vol. 45, no. 12, pp. 4326–4336, 2012. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0031320312002786>
- [42] F. Yuan, J. Shi, X. Xia, Y. Fang, Z. Fang, and T. Mei, “High-order local ternary patterns with locality preserving projection for smoke detection and image classification,” *Information Sciences*, vol. 372, pp. 225–240, 2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0020025516306168>
- [43] S. Frizzi, R. Kaabi, M. Bouchouicha, J.-M. Ginoux, E. Moreau, and F. Fnaiech, “Convolutional neural network for video fire and smoke detection,” in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 877–882.
- [44] Z. Yin, B. Wan, F. Yuan, X. Xia, and J. Shi, “A deep normalization and convolutional neural network for image smoke detection,” *IEEE Access*, vol. 5, pp. 18 429–18 438, 2017.
- [45] K. Gu, Z. Xia, J. Qiao, and W. Lin, “Deep dual-channel neural network for image-based smoke detection,” *IEEE Transactions on Multimedia*, vol. 22, no. 2, pp. 311–323, 2020.
- [46] J. Wang, L. P. Tchapmi, A. P. Ravikumar, M. McGuire, C. S. Bell, D. Zimmerle, S. Savarese, and A. R. Brandt, “Machine vision for natural gas methane emissions detection using an infrared camera,” *Applied Energy*, vol. 257, p. 113998, Jan. 2020. [Online]. Available: <https://doi.org/10.1016/j.apenergy.2019.113998>
- [47] A. Groshenry, C. Giron, T. Lauvaux, A. d’Aspremont, and T. Ehret, “Detecting methane plumes using prisma: Deep learning model and data augmentation,” 2022.
- [48] S. Kumar, C. Torres, O. Ulutan, A. Ayasse, D. Roberts, and B. Manjunath, “Deep remote sensing methods for methane detection in overhead hyperspectral imagery,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1765–1774.
- [49] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, “Deep learning-based classification of hyperspectral data,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2094–2107, Jun. 2014. [Online]. Available: <https://doi.org/10.1109/jstars.2014.2329330>
- [50] W. Zhao and S. Du, “Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach,” *IEEE Transactions*

- on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, Aug. 2016. [Online]. Available: <https://doi.org/10.1109/tgrs.2016.2543748>
- [51] P. Jia, M. Zhang, W. Yu, F. Shen, and Y. Shen, “Convolutional neural network based classification for hyperspectral data,” in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, Jul. 2016. [Online]. Available: <https://doi.org/10.1109/igarss.2016.7730323>
 - [52] U. B. Gewali, S. T. Monteiro, and E. Saber, “Machine learning based hyperspectral image analysis: A survey,” 2018. [Online]. Available: <https://arxiv.org/abs/1802.08701>
 - [53] S. Murala, A. B. Gonde, and R. P. Maheshwari, “Color and texture features for image indexing and retrieval,” in *2009 IEEE International Advance Computing Conference*, 2009, pp. 1411–1416.
 - [54] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
 - [55] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Sep. 2009. [Online]. Available: <https://doi.org/10.1007/s11263-009-0275-4>
 - [56] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jun. 2014. [Online]. Available: <https://doi.org/10.1007/s11263-014-0733-5>
 - [57] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
 - [58] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Apr. 2015. [Online]. Available: <https://doi.org/10.1007/s11263-015-0816-y>
 - [59] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, 2005, pp. 886–893 vol. 1.

- [60] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: <https://doi.org/10.1023/b:visi.0000029664.99615.94>
- [61] X. Peng, L. Wang, X. Wang, and Y. Qiao, “Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice,” *Computer Vision and Image Understanding*, vol. 150, pp. 109–125, Sep. 2016. [Online]. Available: <https://doi.org/10.1016/j.cviu.2016.03.013>
- [62] F. Perronnin, J. Sánchez, and T. Mensink, “Improving the fisher kernel for large-scale image classification,” in *Computer Vision – ECCV 2010*. Springer Berlin Heidelberg, 2010, pp. 143–156. [Online]. Available: https://doi.org/10.1007/978-3-642-15561-1_11
- [63] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: <https://doi.org/10.1145/3065386>
- [64] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014. [Online]. Available: <https://arxiv.org/abs/1409.1556>
- [65] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.
- [66] S. Targ, D. Almeida, and K. Lyman, “Resnet in resnet: Generalizing residual architectures,” 2016. [Online]. Available: <https://arxiv.org/abs/1603.08029>
- [67] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *CoRR*, vol. abs/1311.2524, 2013. [Online]. Available: <http://arxiv.org/abs/1311.2524>
- [68] S. Ren, K. He, R. B. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks,” *CoRR*, vol. abs/1506.01497, 2015. [Online]. Available: <http://arxiv.org/abs/1506.01497>
- [69] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, “Mask R-CNN,” *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: <http://arxiv.org/abs/1703.06870>
- [70] *OGC GeoTIFF Standard*, Open Geospatial Consortium Std., 2023. [Online]. Available: <https://www.ogc.org/standards/geotiff>

- [71] A. Clark, “Pillow (pil fork) documentation,” 2015. [Online]. Available: <https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf>
- [72] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: <https://doi.org/10.1038/s41586-020-2649-2>
- [73] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [74] Rasterio: Geospatial data in python. <https://rasterio.readthedocs.io>. Accessed on 2023-10-23.
- [75] A. D. Snow, J. Whitaker, M. Cochran, J. V. den Bossche, C. Mayo, J. de Kloe, C. Karney, G. Ouzounoudis, J. Dearing, G. Lostis, Heitor, Filipe, R. May, M. Itkin, B. Couwenberg, G. Berardinelli, T. G. Badger, N. Eubank, M. Dunphy, M. Brett, M. Raspaud, M. A. da Costa, K. Evers, J. Ranalli, J. de Maeyer, E. Popov, C. Gohlke, C. Willoughby, C. Barker, and B. M. Wiedemann, “pyproj4/pyproj: 2.6.1 release,” May 2020. [Online]. Available: <https://doi.org/10.5281/zenodo.3783866>
- [76] Geopy: Python geocoding toolbox. <https://geopy.readthedocs.io>. Accessed on 2023-10-23.
- [77] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, pp. 261–272, 2020.
- [78] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, “scikit-image:

- image processing in Python,” *PeerJ*, vol. 2, p. e453, 6 2014. [Online]. Available: <https://doi.org/10.7717/peerj.453>
- [79] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- [80] J. McKeeve and D. Jervis, “Validation and metrics for emissions detection by satellite,” 2022. [Online]. Available: https://go.ghgsat.com/hubfs/Reports/WhitePaper_Validation%20and%20Metrics%20for%20Emissions%20Detection%20by%20Satellite_JMcKeeverD.Jervis_092022.pdf?utm_campaign=Reports%20%26%20White%20Papers&utm_medium=email&_hsmi=229454194&_hsenc=p2ANqtz-9Nhv5tlaVlx2WVeCtK92PcMNkdqeyDSiNOOZTJkx4bHCvzhGk9GyEevi7LPo4BgtCfwA0iUBXmU3g&utm_content=229454194&utm_source=hs_automationhttps://go.ghgsat.com/hubfs/Reports/WhitePaper_Validation%20and%20Metrics%20for%20Emissions%20Detection%20by%20Satellite_JMcKeeverD.Jervis_092022.pdf?utm_campaign=Reports%20%26%20White%20Papers&utm_medium=email&_hsmi=229454194&_hsenc=p2ANqtz-9Nhv5tlaVlx2WVeCtK92PcMNkdqeyDSiNOOZTJkx4bHCvzhGk9GyEevi7LPo4BgtCfwA0iUBXmU3g&utm_content=229454194&utm_source=hs_automation