Final Report

IPSTERS

IPSentinel Terrestrial Enhanced Recognition System

João Fonseca

Advisor: Prof. Fernando Lucas Bação, PhD

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

May 14, 2020

Abstract

This report documents the work developed towards the research project "IPSTERS - IPSentinel Terrestrial Enhanced Recognition System". It focuses on the exploration of several machine learning (ML) techniques, covering different stages of a Land Use/Land Cover Classification (LULC) pipeline. These techniques aim to minimise problems typically found in this kind of data, namely data ingestion, feature selection, data filtering and classification.

Contents

List of Figures

List of Tables

1. Introduction

The Copernicus programme is the European Union (EU) Earth Observation (EO) programme, headed by the European Space Agency, and the developer of the Sentinels EO satellites. The IPSentinel is the Portuguese infrastructure developed by Direção Geral do Território (DGT) and Instituto Português do Mar e da Atmosfera (IPMA) for storing and providing images of Sentinel satellites, covering the Portuguese territory and its search and rescue area. This free EO data has been used to inform environmental models, business strategies and political decisions. However, this ever growing volume of data requires big data workload that is overwhelming for Public Administration (PA) agencies. As a result, the use of IPSentinel data has not been widely adopted by the PA that would profit from it.

Often what these agencies need for their goals are digested data in the form of specific class maps. These value-added products are often called level-3 products and are fundamental for land-management and for the country's international commitments such as the estimation of CO2 emissions. These products are mainly obtained by visual interpretation of high resolution satellite imagery, requiring significant allocation of human resources from the PA and taking a long time to produce, being one of the reasons for the low update rate and low resolution. In the case of COS (Portuguese Land cover-land use maps) they are produced every 5 years with 1ha resolution and EU CORINE maps at least every 6 years with 25ha.

1.1 Purpose and Objectives

The main goal of this project is to explore the applications and limitations of artificial intelligence (AI) algorithms with accelerated processing hardware capabilities, as a unit of the IPSentinel for the digestion of large volumes of remotely sensed data, to produce level-3 products for land applications with the least amount of human intervention. We propose exploring two artificial intelligence approaches, one applying active learning techniques and another based on fuzzy logic.

Specifically, this report focuses on the exploration of techniques to improve the quality of LULC map outputs by the model under development. As such, topics such as Dimensionality Reduction, Anomaly Detection and Classification methods will be discussed.

Most of the work developed so far (excluding presentations and all preliminary experiments) is available in the Github repositories of the project:

1. **remote-sensing:** Includes all experiments except data filtering/anomaly detection methods.

2. **data-filtering:** Includes experiments ran for data filtering/anomaly detection methods.

1.2 Document Structure

This document is organised in 4 chapters. Chapter 2 analyses the state of the art on dimensionality reduction, data filtering and classification methods. Chapter 3 provides a brief overview of the datasets used, preprocessing methods experimented and classification methods tested. Chapter 4 presents the results obtained for each of these topics.

2. Literature Review

The state of the art on the main challenges identified for the project is shown here. The multispectral imagery used in this project is targeted for a large area (continental Portugal) and contains complex and highly correlated spectral information. Although the presence of multicollinearity doesn't pose a problem for the adequacy of modern ML models to predict a target variable [4], high dimensional data is difficult to process and therefore strains the capacity of producing accurate LULC maps. Dimensionality reduction techniques are used to address such an issue. These techniques allow the selection of the most important feature within the image composites used, allowing for 1) a clearer understanding of the most important features for LULC classification, 2) accelerated model training and 3) avert the curse of dimensionality [6].

The datasets used in this project are Sentinel 2 image composites overlaid with the Portuguese Land cover-land use maps (COS). This type of data is prone to noisy labelling for various reasons. Label noisy may come from discrepancies between the original map's resolution (in the case of COS, 1ha) and the objective map's resolution (for this project, $10m^2$), as well as land cover type changes or classification errors. Mislabeled points are potentially problematic since the use of thesepoints for producing land cover maps at a $10m^2$ resolution can output generalised and/or erroneous predictions. Therefore, it is necessary to explore the need for noisy data filtering.

Finally, classification algorithms are discussed in the last section. These algorithms should be resistant to noise and/or have correcting measures to decrease the effect of the noisy data in the model's learning phase.

2.1 Dimensionality Reduction Methods

Dimensionality reduction techniques can be divided into 2 parts, feature selection and feature extraction, both techniques explored below. Figure 2.1 depicts the different types of techniques to reduce a dataset's dimensionality.

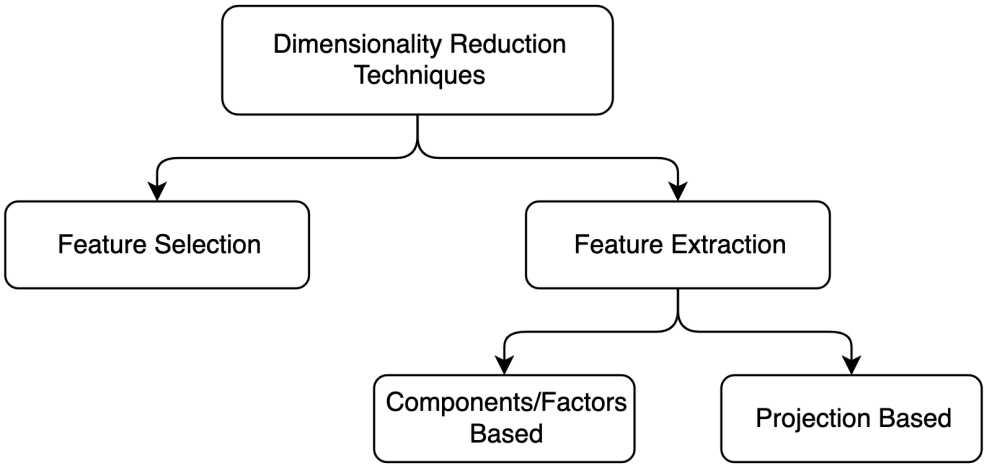


Figure 2.1: Dimensionality Reduction Techniques

Although all these methods reduce the dimensionality of data, the means used to do so varies.

Feature Selection

Feature selection methods reduce dimensionality through the selection of the most important features among the ones already existing in the dataset. The criterion used either improves, maintains the model accuracy, or simplifies the model complexity. Below are presented commonly used supervise feature selection methods. More information on listed and/or additional methods can be found in [3, 6].

1. Correlation based feature selection (CFS). It's based on the predictive power of a feature subset, found by computing the correlation between the feature subset and the target feature. It is therefore a generalisation of the Correlation Criteria as defined in [6].

2. ReliefF [12]. It's an extension of Relief [11] to support multi-class problems. For each instance, the algorithm considers the closest instance from each class to update the feature score vector. The score of any given feature decreases if it differs more in nearby instances of the same class than nearby instances of the other class, and increases in the reverse case.

3. Random Subspace Method (RSM) and Random Forest Method (RFM). Consists in the training of base classifiers on different feature subsets. These methods use the trained classifiers to assess each feature's importance to the prediction of the target variable. In the case of RFM, it is important to consider that Tree based classifiers are trained through the minimisation of entropy on each data split. As such, variables used earlier for decision rules along a decision tree's path is deemed as having a higher importance than the remaining variables.

Along with the presented methods, a number of variations are being applied by practitioners, such as Permutation Feature Importance (shuffling each feature after training a classifier and check accuracy drops) and Drop Column Feature Importance (dropping one feature and checking differences in accuracy compared to a classifier using all available features).

2.2 Data Filtering Methods

The accuracy of a classifier is directly affected by the quality the training data used [1]. Despite the importance of classification tasks in remote sensing, obtaining well-labelled data is time consuming, impracticable and expensive [20]. To the best of our knowledge there is no systematic review on data selection/filtering methods for the remote sensing domain. Through the analysis of the different algorithms available, 2 distinct types of filtering were identified, each dividing into 2 subgroups. Figure 2.2 depicts the different methods found for this purpose.

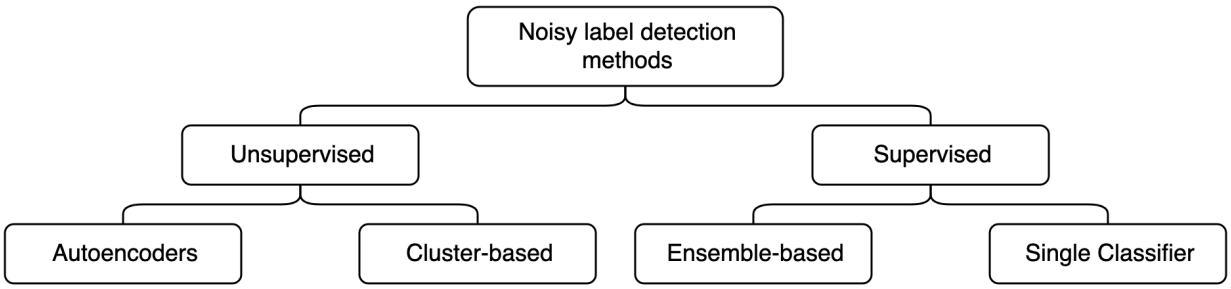


Figure 2.2: Different types of methods found for noisy label detection.

LULC maps are commonly presented in a shapefile format. A recent study employed polygon membership information for each pixel to detect labelling errors within each polygon. This was done through the use of clustering methods and cross cluster consistency measurement methods [18]. Variations on this method are explored and documented below.

A frequent type of approach to address this challenge is based on the use of different machine learning classifiers to detect these errors [2, 9, 14, 23, 26, 20, 5, 1, 24]. These methods benefit from classifiers' specificities, especially Random Forest, to implement different voting strategies and model fitting methods to improve label noise detection. Since these methods don't use domain specific information, they can be used over any machine learning problem.

Although the impact of these filtering methods on robust classifiers is unknown, some studies document the impact of label noise on classification accuracy, showing the robustness of Random Forests and Deep Learning algorithms [19, 21]. Relevant filtering methods presented in this section will incorporate the experiment, where the impact of data filtering can be assessed (although these results are accompanied with significant limitations).

Unsupervised methods

The best performing label noise detection algorithm based on clustering methods was proposed by [18]. This method assumes the availability of a LULC map provided at a polygon level. The training set identification method is based on 3 steps:

1. Polygon k-means clustering. Starts by clustering the points within each separate polygon using the K-Means clustering algorithm and the Calinski Harabasz index (CHi) to automatically determine the optimal number of clusters for each polygon. The majority cluster of each polygon passes on to the next step, while all pixels belonging to the minority clusters are dropped and disregarded for potential training set candidates. Figure 2.3 depicts a qualitative example of this step.

2. Polygon consistency analysis. Majority clusters are submitted to consistency analysis, which is done by computing each cluster's Bhattacharyya distance (B-distance) to compute the similarity of each subset to the entire set of clusters of the corresponding class. Afterwards, the clusters with distance from the entire set of clusters smaller than the 65^{th} percentile of the cluster distances are selected for Stratified Random Sampling.

3. Stratified Random Sampling. This step ensures the generation of a balanced dataset, proportionate to the original class ratios.

Figure 2.3: Qualitative example of polygon k-means clustering result. (a) Original polygon associated to the "crop" label. (b) Dominant land-cover class detected C1. (c) First minor class detected C2 (road). (d) Second minor class detected C3 (river). Source: [18]

Through qualitative analysis, this method achieved satisfactory results. Regardless, it is difficult to determine the quality of these results without reference data.

Other approaches include the use of autoencoders. Each class is used to train an autoencoder, allowing the models to learn how to make latent representations of instances belonging to a single class. For each instance, a reconstruction error will be computed, using it as the mislabel score (the higher the error, the more likely the instance is mislabeled) [24].

Supervised methods

All supervised methods take a quantitative approach to the problem. Typically, these methods minimise either one of two errors. On the one hand, instances incorrectly tagged as mislabeled represent an E1 error. On the other hand, mislabeled instances left untagged represent an E2 error. Figure 2.4 shows a visual representation of the two types of error.

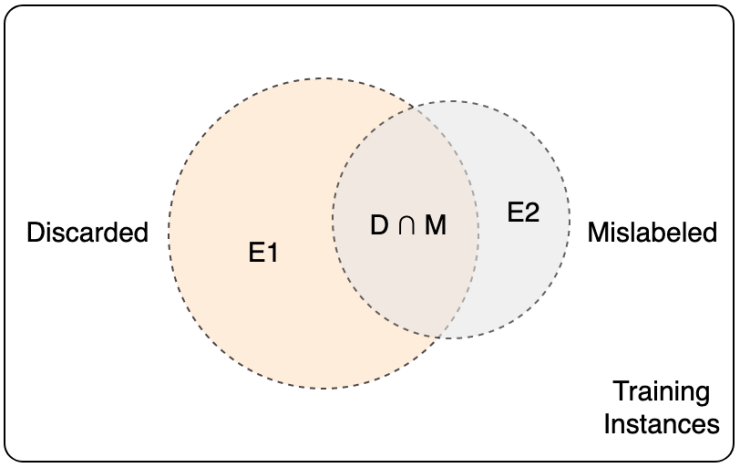


Figure 2.4: Types of detection errors. Adapted from [2].

Supervised methods can be divided into 2 subgroups:

1. Single model approaches. They use a single model to determine wether a specific instance is correctly labelled. A common implementation includes the use of a standard classifier to predict the label of an instance and tagging it as mislabeled if the classifier's prediction differs from the original label [2].

2. Ensemble-based approaches. They use a set of weak classifiers (most commonly Decision Trees) to generate a number of output predictions. These classifiers are always trained in a subset of the dataset, which are then used to predict the label of the remaining dataset. Consequently, this set of classifiers is trained k times, depending on the number of splits used in the data stratification. These predictions are then changed to a binary output of correct/incorrect prediction according to the original label. The more incorrect predictions over a single instance there are, the more likely it is for it to be tagged as mislabeled. The methods for aggregation of the set of individual votes to compute the final score differ according to the method used.

Single model approaches benefit from reduced computation power and generally improve the quality of the training set generated, although they rarely outperform any ensemble-based approach [2, 5, 1, 20, 26, 23].

Within ensemble-based approaches there is a simple underlying calculation of a mislabel likelihood (henceforth referred to as mislabel ratio). For traditional methods the calculation consists in a simple ratio between the count of incorrect prediction over the total number of predictions:

$$MR(x) = \frac{n - \sum_{i=1}^{n} v_i^y}{n} \tag{2.1}$$

Where n is the number of predictions for an instance x and v_i^y is 0 if the predicted label differs from the true label, 1 otherwise. Although any threshold could be set for MR in order to consider an instance's label incorrect, most of the literature uses 2 different thresholds, referred to as consensus ($MR = 1$, i.e., all predictors fail to classify an instance's true label) and majority filtering ($MR \geq 0.5$, i.e., the majority of predictors fail to classify an instance's true label). Therefore, consensus filters focus on the minimisation of E1 error whilst majority filtering minimises E2 error [2, 23]. Other methods have been used to measure mislabel likelihood. [1] proposed an ensemble margin score, defined as:

$$EM(x) = \frac{2\sum_{i=1}^{n} v_i^y - n}{n - \sum_{i=1}^{n} v_i^y} \tag{2.2}$$

Both majority and consensus filtering have been used in a number of recent publications [5, 1, 20, 26, 23]. Some of these methods [1, 23] achieved improved performance by implementing an iterative approach to the problem, consisting in multiple applications of filtering process a dataset, gradually reducing its size on each iteration. Figure 2.5 provides a visual representation of an ensemble-based filtering pipeline.

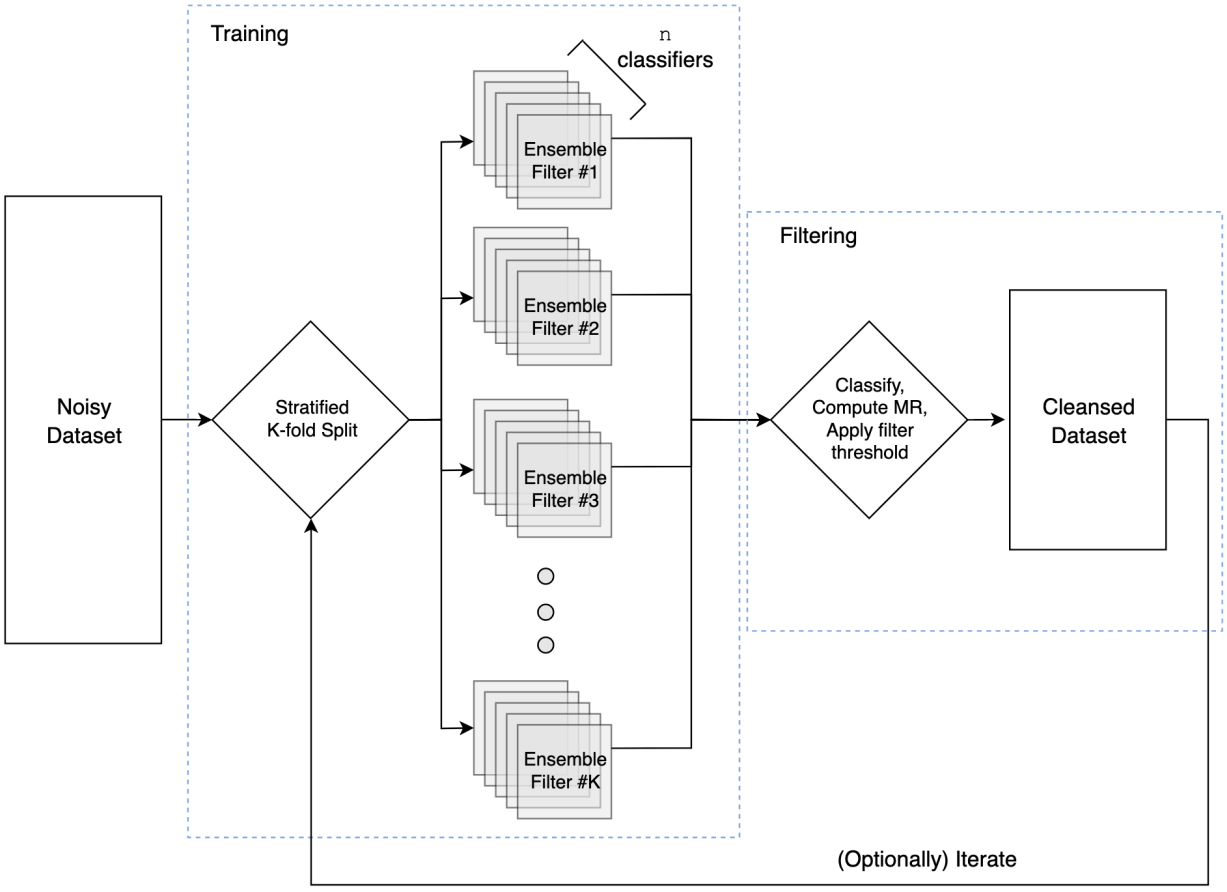
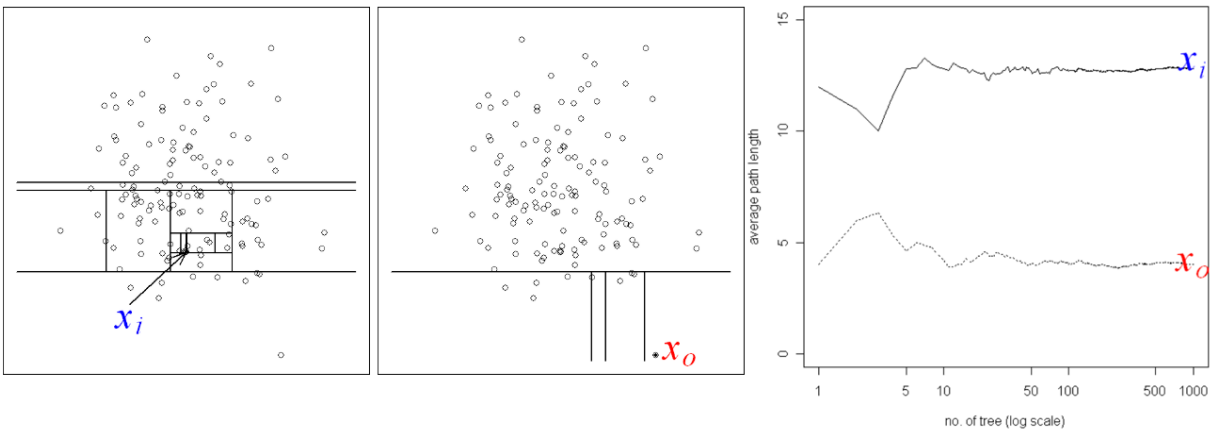


Figure 2.5: Ensemble-based approaches pipeline.

Isolation Forest (iForest) [14] is another common method used for data filtering. It benefits from the use of a set of Isolation Trees (iTrees) that attempt to isolate each instance within the dataset. The rationale behind this algorithm consists in the measurement of the average path length an instance goes through in all iTrees, assuming that an anomaly (in this case, a mislabeled instance) has a different profiles than that of the remaining population of the corresponding label. Consequently, a mislabelled instance is expected to have a smaller path length than that of a correctly classified instance, as demonstrated in Figure 2.6.



(a) Isolating x_i (b) Isolating x_o (c) Average path lengths converge

Figure 2.6: Anomaly detection using the iForest algorithm. Anomalies are more susceptible to isolation and hence have short path lengths. (a) a normal point x_i requires twelve random partitions to be isolated; (b) an anomaly x_o requires only four partitions to be isolated. (c) averaged path lengths of x_i and x_o converge when the number of trees increases. Source: [14]

2.3 Classification Methods

The most common classifiers identified for land cover image classification are Random Forests (RFC), Support Vector Machines (SVM), K-Nearest Neighbours (KNN), artificial neural networks (ANN), Maximum Likelihood (ML) and Decision Trees (DT) [10, 17]. However, recent literature has focused on the implementation of robust classifiers such as Convolutional Neural Networks with varying structures, Random Forests and Autoencoders [22, 25, 13].

Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a type of deep learning networks, commonly applied in image data. Figure 2.7 shows a visual representation of the basic mechanisms within a typical CNN. This type of classifier has been extensively used for LULC classification and has achieved state of the art results in reference datasets such as Indian Pines.

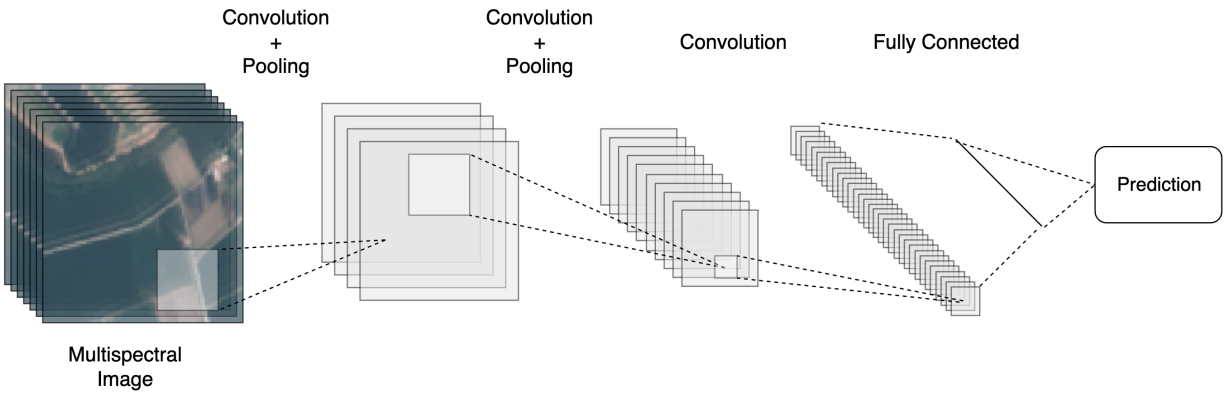


Figure 2.7: Typical Convolutional Neural Network example.

A particular CNN implementation specific to multi/hyper-spectral imaging classification is the Hybrid Spectral Net [22]. This CNN consists of a set of three 3D convolutions (as opposed to the conventional 2D convolutions), performing both Spectral and Spatial feature learning, and one 2D convolution, followed by two fully connected hidden layers. This model achieved state-of-the-art performance on the Indian dataset and was one of the first models to implement a CNN architecture based on both 3D and 2D convolutions.

Another CNN implementation is the Residual Neural Network (ResNet) [7]. Its architecture allows the construction of very deep networks while circumventing the so-called vanishing gradient problem. This algorithm along with other deep neural networks have been tested for land cover classification, achieving state-of-the-art classification accuracies [16].

Random Forests

Random Forest Classifiers were originally proposed by [8]. RFCs are an ensemble learning method that combine a set of decision trees, each using a random subspace of the original feature set. Random Forests correct the Decision tree's tendency to overfit by using the mode of the prediction of individual trees as its final prediction.

3. Methodology

This chapter describes the work developed, along with the models implemented for each studied step of the pipeline. The implementation of the experimental procedure was based on the Python programming language, all functions, algorithms, experiments, and results reported are available in the Github repository of the project.

3.1 Datasets

For the experiments reported four datasets are used:

1. LUCAS dataset. The 2015 LUCAS data was used as reference data for both model training and validation. The LUCAS point label represents the corresponding land cover/use type within the radius of 1.5 m for homogeneous classes and a 20 m radius extent ("extended window") for heterogeneous classes (e.g., shrubland), gathered by field observation and a very high-resolution photo interpretation [15]. In order to reduce the risk of having Landsat pixel information represented wrongly in the field, we only kept points observed in-situ from a close distance (<100 m). With the same objective we removed the points which had linear features in the observation (e.g., roads). This procedure was not solely applied to the class of "artificial land," as this would remove most parts of the samples. Furthermore, points with cloudy pixels in the Landsat data were also excluded. This way, 1694 out of 2060 LUCAS points were retained. This dataset contains eight classes that represent the main land cover types for the study area. This pixel selection excluded a large number of unacceptable reference points, and we assumed the remaining ones to be suitable enough to represent the land cover type in a Landsat pixel coverage area of 30x30 m. Further, we surmised that classifiers are capable of overcoming the noise caused by pixels having mixed land cover representation if such pixels are still available in the dataset.

2. COSsim polygons dataset. It uses a Sentinel 2 composite image along with COSsim+2018. This dataset was produced by taking a sample of polygons from each class with a negative buffer applied and extracting the pixels within it.

3. Setubal dataset. Corresponds to sampled points out of Sentinel 2 images and the labels were extracted from the EU's agricultural survey.

4. Coimbra dataset. It consists of a Sentinel-2 Satellite tile (29TNE) from August 2015 to which COS 2015 was overlaid, the study area covers a region around Coimbra, Portugal (see Figure 3.1). Since the size of the entire tile is too large to be processed in a timely manner, a subspace of the image will be used. Figure 3.2 shows the train (top row, outer non-highlighted area) and test (bottom row) areas along with its true colour image (left column) and ground truth (right column) used to assess the accuracy on each classifier. As such, any pixel and/or neighbouring pixels used for training will never appear in the test set, making them distinct sets.

5. COSsim sampled dataset. Corresponds to sampled points out of a Sentinel 2 composite image along with COSsim+2018 using the stratified sampling approach.

6. COS sampled dataset. Consists of a stratified sample of a Sentinel 2 monthly composite image (between March 2018 and February 2019) of the tile 29SNC along with COS 2015 and 2018 to allow for cross temporal analyses and land cover change analyses. The cross temporal analysis was done with a stratified sample of a similar composite for the period of February 2019 and January 2020. Before the sampling phase, cloud masking was applied to remove the values on months/pixels covered by clouds. These values were then imputed using the linear interpolation of values on adjacent months.

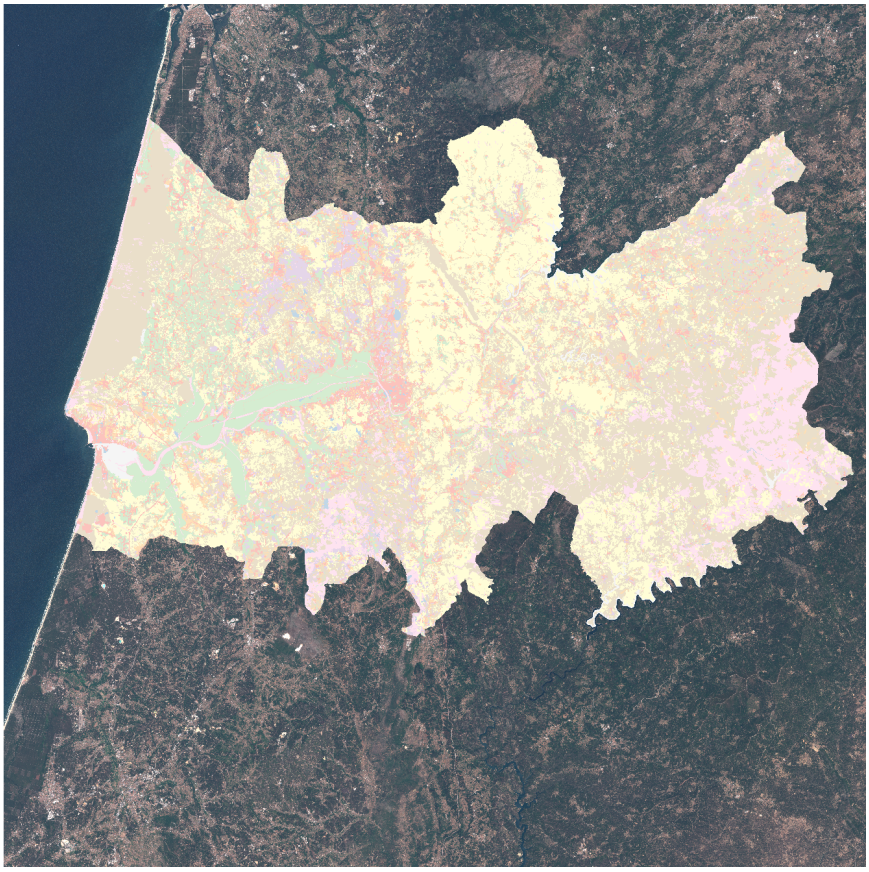


Figure 3.1: Tile 29TNE.

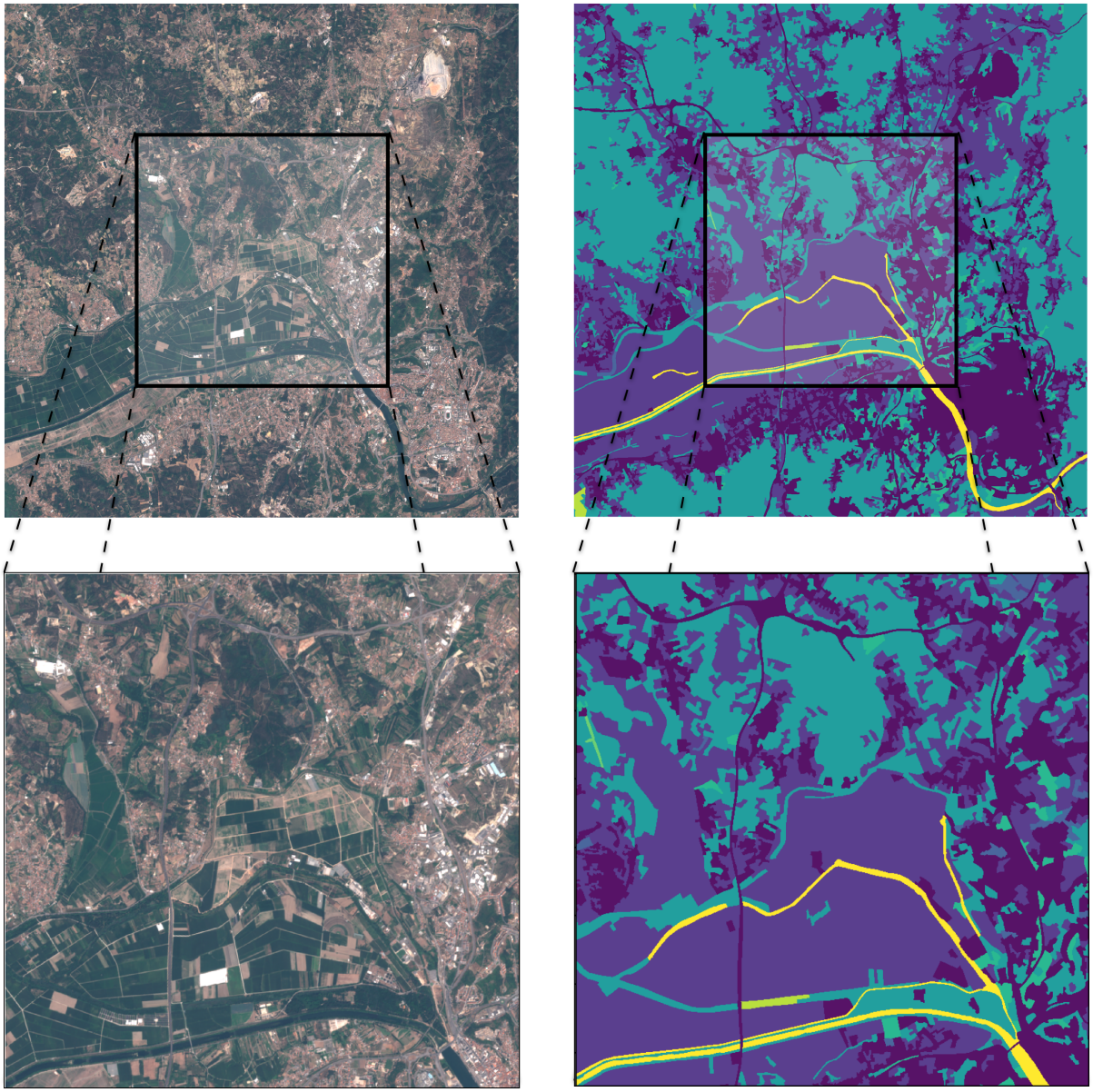


Figure 3.2: Coimbra dataset. The top row, outer non-highlighted area is the train set, while the bottom row area is the test set. The left column shows their true colour image and the right column shows the ground truth.

3.2 Preprocessing

In order to select the pixels for the COSsim polygons, COSsim sampled and COS sampled datasets, a negative buffer was applied on labelled polygons in order to avoid border pixels, which are frequently mislabeled. Afterwards, a set of experiments was conducted separately for each different topic of interest.

An experiment will be run in order to test different feature selection methods. The methods applied are the Random Forest Method, Permutation Feature Importance and the intersection of both. An Overall Accuracy will be provided after training a RFC with each method's selected features.

A set of filters will be tested on the LUCAS dataset. In order to check each method's robustness to noise detection, different levels of noise is introduced for each test. The noise is introduced based on combinations of labels that are difficult to distinguish by a classifier. This is done through the training of a Random Forest on the original dataset and extracting these label pairs from the confusion matrix. This way, we avoid random noise and approach the type of noise that could be found in a real scenario.

An additional filtering method will be tested, combining clustering and ensemble methods. As a first stage, en ensemble filter composed of two RFC with 10 and 25 estimators, a logistic regression, a single Decision Tree and a Multilayer Perceptron Classifier is used. Simultaneously, a Minibatch K-Means clustering algorithm is run for each class. The number of K is calculated using a granularity measure:

$$K(granularity) = int(granularity\sqrt{size(X)}) \tag{3.1}$$

Where *granularity* is a hyperparameter and X is the feature matrix. Afterwards, the average filtering results are computed per cluster and the top $n\%$ mislabeled clusters per class are discarded from the training dataset. This method also ensures that at least one cluster per class is kept and the threshold to define n can be done either through a threshold for the mislabel rate or a threshold for the minimum percentage of observations to be kept.

Additionally, for traditional methods like majority and consensus filtering the number of K splits is set up as a hyperparameter, as opposed to the predefined $K = 4$ used in the literature.

3.3 Classification

The classifiers tested include the ones listed in the literature review: Random Forest Classifier, Hybrid Spectral Net, fully connected autoencoder and ResNet50 (with 2 fully connected layers, a total of 52 hidden layers).

4. Results

4.1 Preprocessing

4.1.1 Dimensionality Reduction

The methods applied are described in section 2.1. The Gini method is the calculation of the importance of a feature using the (normalised) total reduction of the Gini criterion brought by that feature. Permutation is the shuffling of a feature after training a classifier and measuring the accuracy difference, returning the most important features as the ones that cause the highest accuracy drop. The Intersect method is the intersection of both methods, maintaining only the features that are selected with both methods. The results of the experiment were obtained by training a RFC using the selected features for each method and are presented in Tables 4.1 and 4.2.

Selection Method	Nbr of features used	Accuracy	F-Score	G-Mean
Original	180	0.755	0.761	0.862
Gini	121	0.750	0.758	0.861
Permutation	58	0.749	0.760	0.863
Intersect	45	0.741	0.752	0.858

Table 4.1: Feature selection results using the Setubal dataset. The number of features selected are based on optimal results of each selection method.

Method	Accuracy	F-score	G-mean
Original	0.674	0.578	0.713
Correlation Based	0.665	0.569	0.706
Permutation	0.673	0.579	0.713
RFEntropy	0.673	0.580	0.715
RFGini	0.671	0.576	0.712
ReliefF	0.679	0.580	0.715

Table 4.2: Feature selection results using the COS sampled dataset. The number of features selected are based on optimal results of each selection method.

Another two different dimensionality reduction experiments were done for the COSsim sampled and COS sampled datasets, using the same procedure. The feature rankings for both datasets are available as csv files. Figures 4.1 and 4.2 show the evolution of the different evaluation metrics on varying number of features using the ReliefF method on the COSsim sampled and COS sampled datasets.

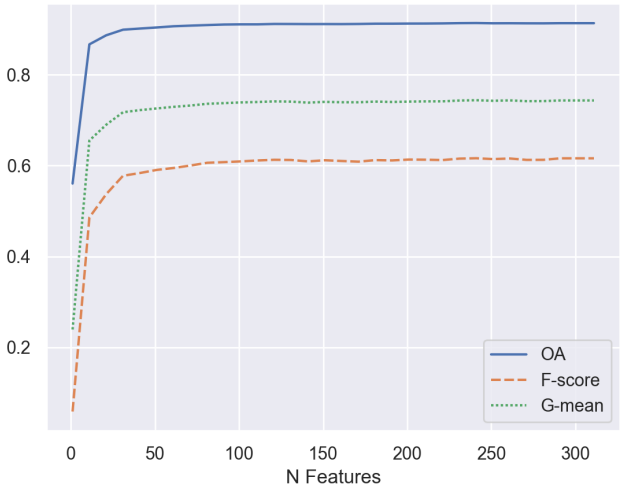


Figure 4.1: Evolution of the different scorers on varying number of features using the COSsim sampled dataset and the ReliefF method.

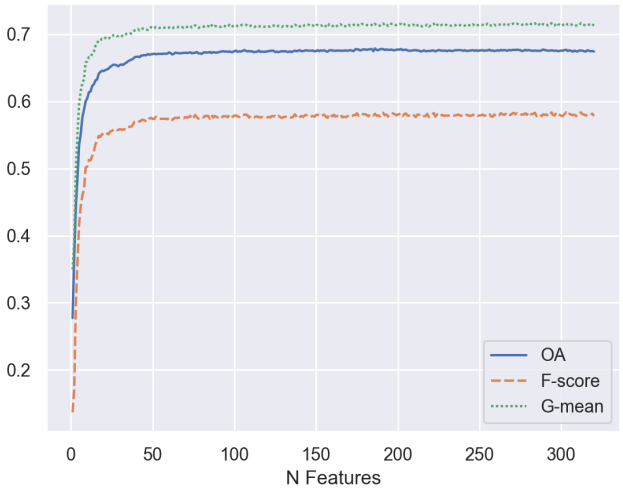


Figure 4.2: Evolution of the different scorers on varying number of features using the COS sampled dataset and the ReliefF method.

Additional analyses based on the COS sampled dataset are presented in tables 4.3 and

4.4 and figure 4.3. Note that negative normalized mean scores do not necessarily mean a negative contribution to the dataset's class separability, but rather a below average contribution.

Band	Normalized Score
NDWI	0.651
NDMI	0.528
01	0.505
09	0.308
02	0.281
06	0.263
03	0.218
NDBI	0.153
8A	0.081
12	0.0
07	-0.027
08	-0.032
11	-0.05
05	-0.086
04	-0.12
NDVI	-0.178

Table 4.3: Normalized mean importance per band.

Month	Normalized Score
1	1.067
11	0.96
4	0.516
10	0.109
9	0.038
5	-0.014
6	-0.034
12	-0.054
8	-0.107
2	-0.14
3	-0.232
7	-0.238

Table 4.4: Normalized mean importance per month.

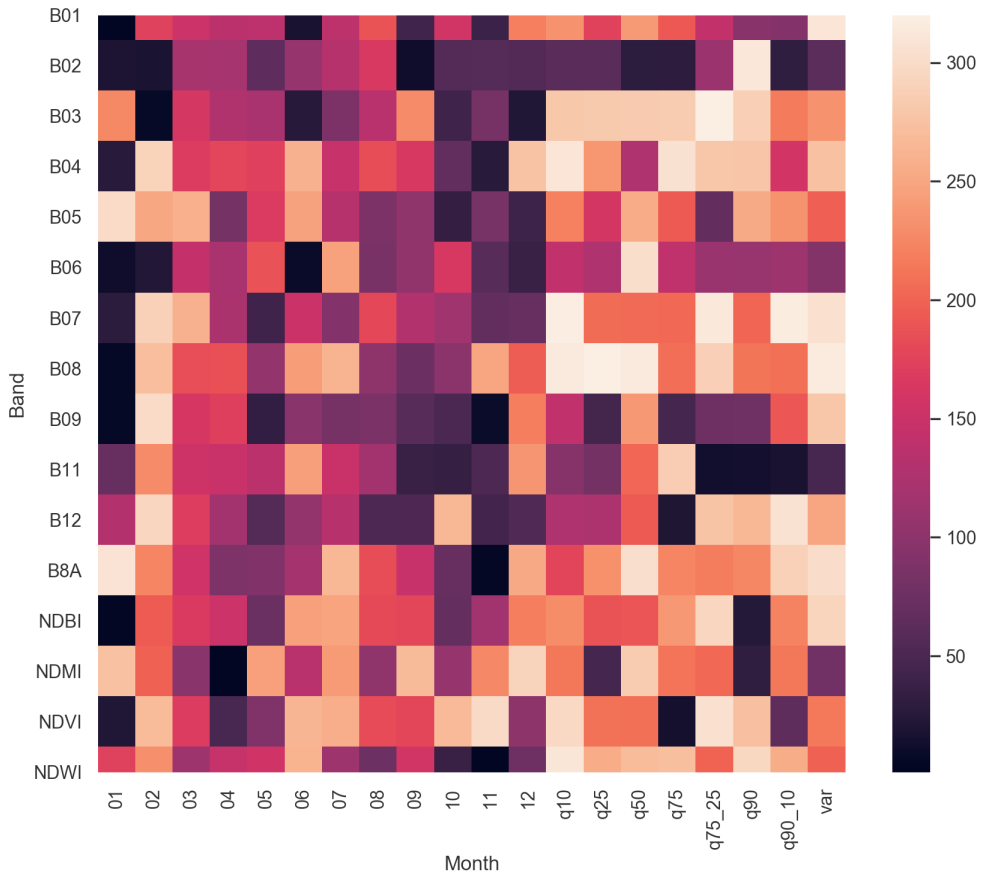


Figure 4.3: Heatmap based on feature rankings.

Figures 4.4 and 4.5 show the most and least important feature's class distribution using kernel density plots.

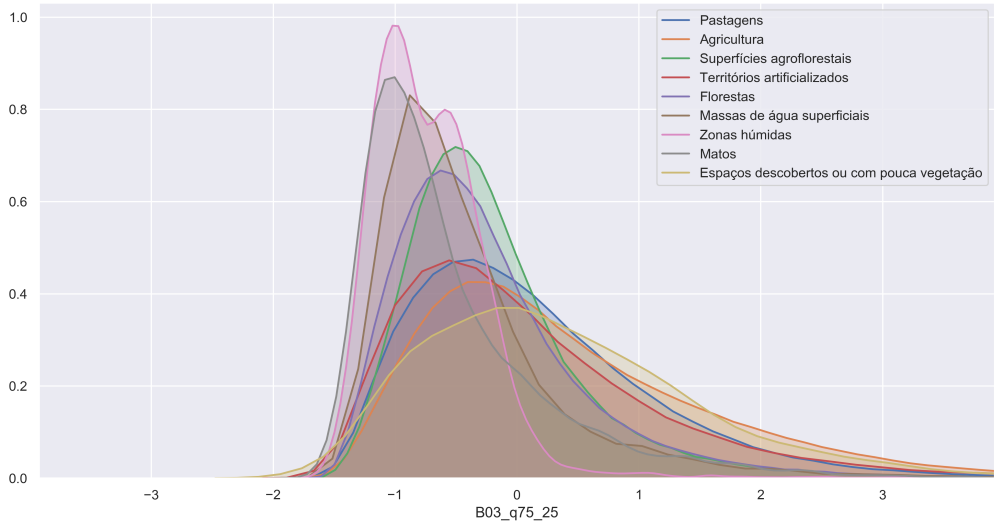


Figure 4.4: Kernel density plot of the feature with lowest importance score.

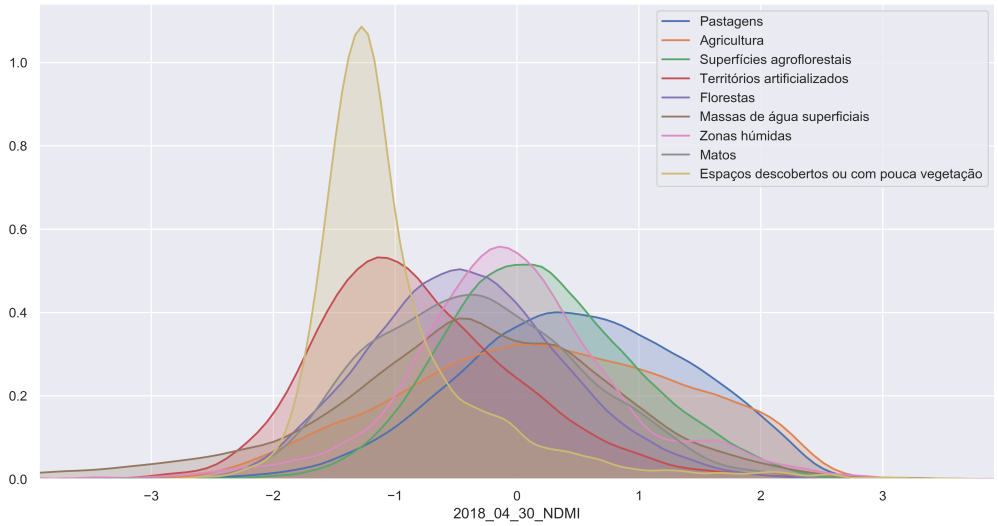


Figure 4.5: Kernel density plot of the feature with highest importance score.

4.1.2 Data Filtering

Figure 4.5 shows the Overall Accuracy of a RFC trained with filtered datasets using each of the filters presented. Note that all methods are tuned for optimal performance on the

given dataset, as opposed to the fixed parameters commonly used in majority, single and consensus filters.

Method	0	5	10	20	30	40
No Filter	0.621	0.622	0.622	0.604	0.559	0.487
Single	0.604	0.604	0.612	0.591	0.574	0.533
Majority	0.619	0.614	0.616	0.599	0.584	0.544
Consensus	0.621	0.627	0.622	0.613	**0.585**	0.534
Our method 1	0.630	0.628	0.624	0.615	0.579	0.538
Our method 2	**0.631**	**0.630**	**0.629**	**0.619**	0.581	**0.552**

Table 4.5: Data filtering results using the LUCAS dataset. Column labels represent the different levels of noise introduced in the original dataset.

A similar experiment was performed on the COS sampled dataset. This experiments uses the same methods, although the dataset is used as-is (i.e., no noise was introduced into the dataset). The results of that experiment are reported in table 4.6.

Method	Accuracy	F-score	G-mean
None	0.673	0.575	0.711
Single	0.652	0.541	0.687
Consensus	0.666	0.563	0.703
Majority	0.655	0.546	0.691
iForest	0.653	0.536	**0.713**
OwnMethod1	0.673	**0.578**	0.712
OwnMethod2	**0.674**	**0.578**	**0.713**

Table 4.6: Data filtering results using the COS sampled dataset.

Aside from the methods presented above, other variations of [18] were experimented. These variations considered different clustering methods and used two step selection processes, as shown in figure 4.6. Clustering methods shown in the columns represent phase 1, which is the initial clustering of the pixels in each polygon. The centroids of these clusters are then clustered in phase 2, using either method presented in the rows. The pixels selected for posterior work are the ones belonging to the subclusters (obtained in phase 1) that belong to the majority cluster (computed in phase 2). This process is done for each existing class in the target variable and the optimal number of clusters in both phases is determined using the Calinski-Harabasz index. The only two situations where such procedure doesn't apply are the methods using Bhattacharyya distance in phase 2, where the same procedure as presented in [18] was used.

		Polygon-Level clustering (phase 1)	
		SOM	K-Means
Consistency Analysis (phase 2)	SOM	✓	✓
	K-Means	✓	✓
	Hierarchical		✓
	Bhattacharyya	✓*	✓**

Figure 4.6: Paris method variations tested. Legend: * Includes per-polygon minority clusters rejection. ** Paris et al. 2019 implementation.

Although some hierarchical clustering methods were experimented, the results were rarely satisfactory. For that reason, these algorithms were excluded for further analysis. Figure 4.7 present the pixel selection outcome for a polygon belonging to the rainfed class.

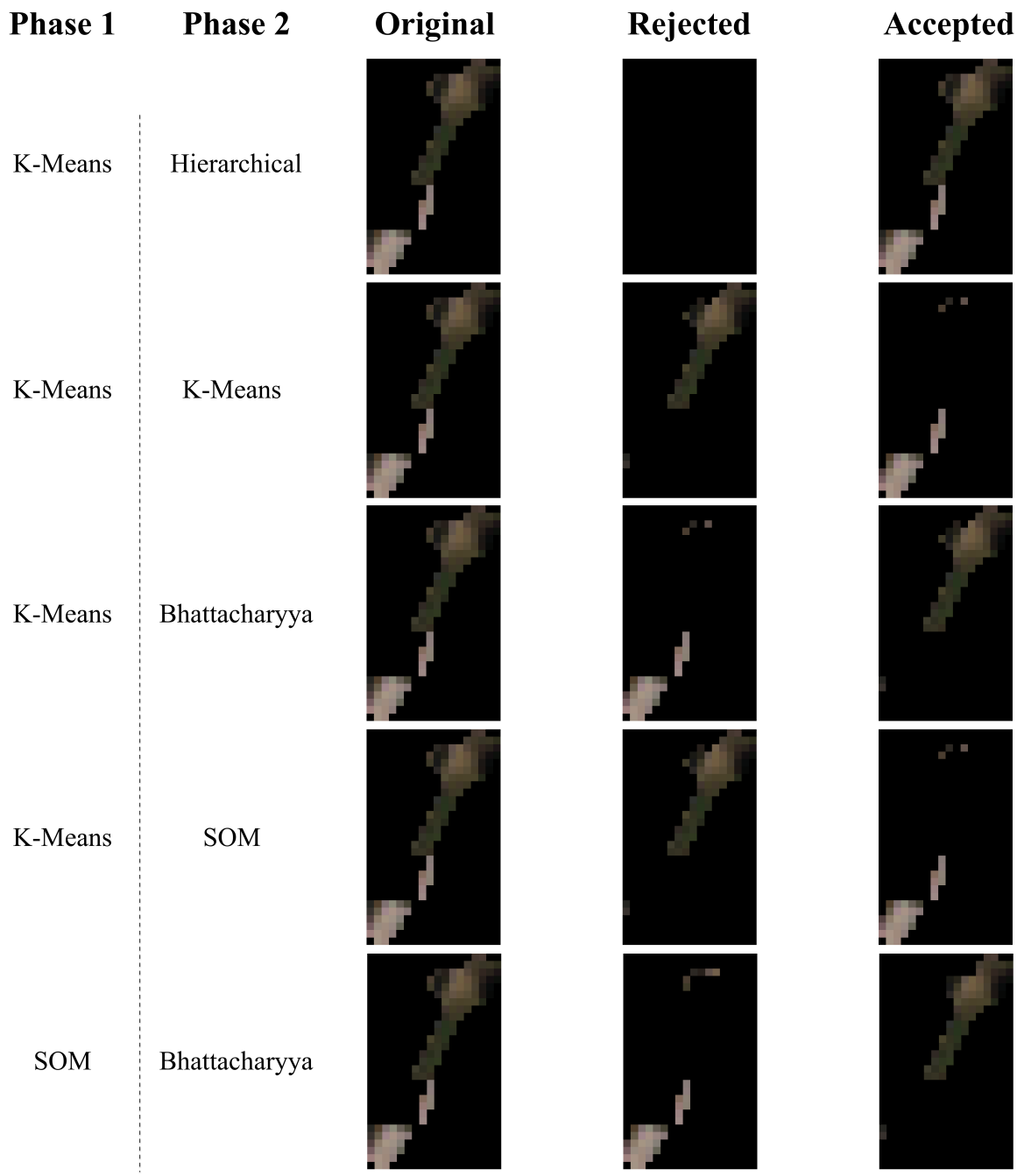


Figure 4.7: Example of Paris variations pixel selection outcomes using a polygon belonging to the rainfed class.

4.2 Classification

The Overall Accuracy (OA), F-Score and Geometric mean (G-Mean) for each classifier's prediction on the test area is shown in table 4.7. Classification cores on the COS sampled dataset is shown in table 4.8

Score	RF	HSN	MLPAE	ResNet50
OA	0.674	0.700	0.717	**0.762**
F-Score	0.354	0.378	0.318	**0.422**
G-Mean	0.559	0.573	0.540	**0.628**

Table 4.7: Classification scores using the classifiers fitted on the training area (Coimbra dataset) with predictions made on the test area, shown in Figure 3.2. Legend: Random Forest Classifier (RF), Hybrid Spectral Net (HSN), Autoencoder-based multilayer perceptron (MLPAE), Residual Neural Network with 50 layers and 2 fully-connected layers (ResNet50).

Score	ABC	GBC	LR	RF
OA	0.506	0.672	0.639	**0.715**
F-Score	0.248	0.596	0.538	**0.646**
G-Mean	0.499	0.73	0.691	**0.757**

Table 4.8: Classification scores using the COS sampled dataset with 5-fold cross validation. Legend: AdaBoost Classifier (ABC), Gradient Boosting Classifier (GBC), Logistic Regression (LR), Random Forest Classifier (RF).

The maps generated by each classifier is displayed in Figure 4.8.

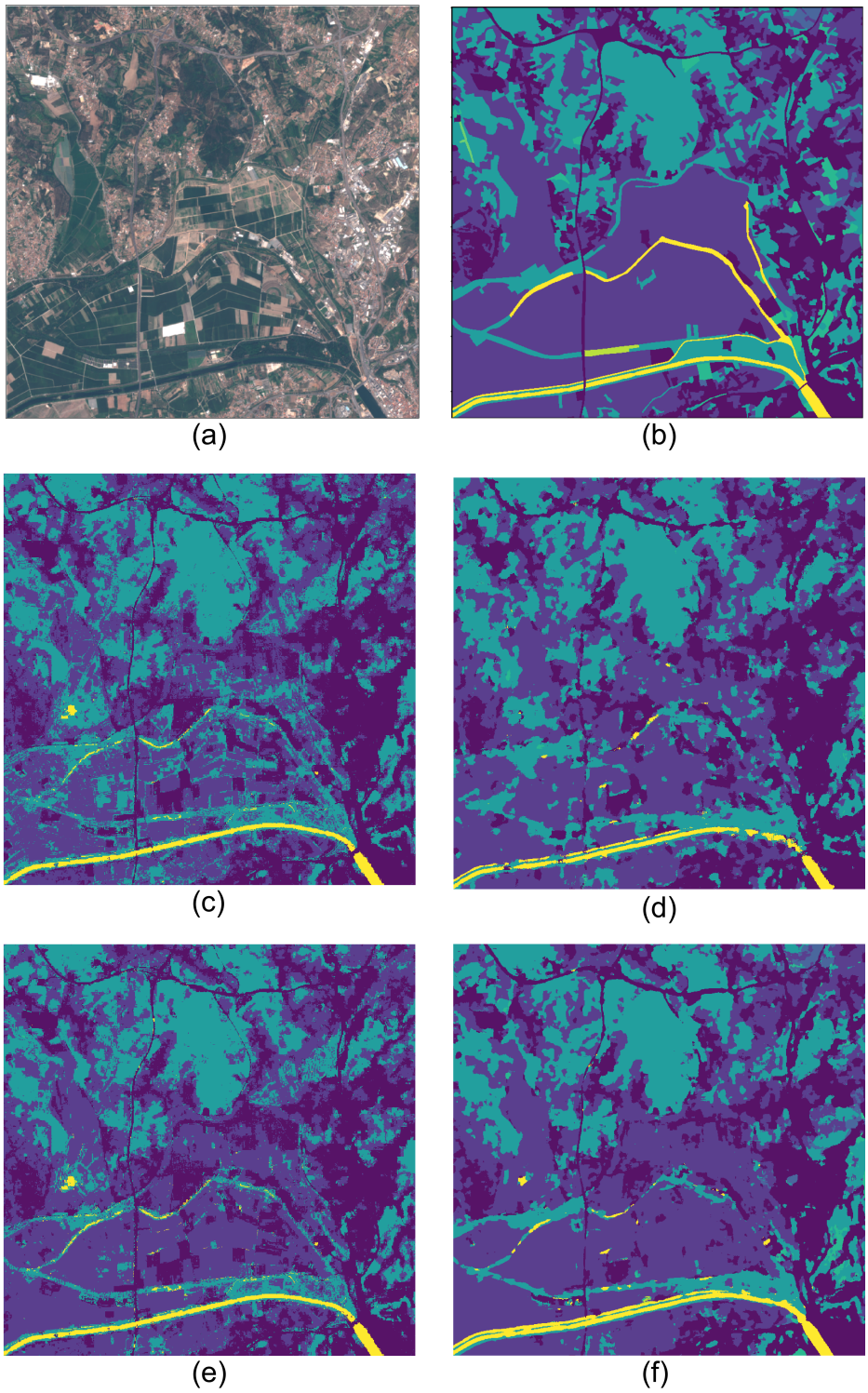


Figure 4.8: Generated maps per classifier. (a) True color image; (b) Ground truth; (c) Random Forest Classifier; (d) Hybrid Spectral Net; (e) Autoencoder-based multilayer perceptron; (f) Residual Neural Network.

Cross temporal validation

Based on the results reported in the previous sections, we selected the best performing methods from each step of the pipeline and measured their performance in a cross temporal setting using the COS sampled dataset. Table 4.9 shows the algorithms used for this step.

Step	Algorithm	Observations
Feature Selection	ReliefF	Reduced dimensionality to the top 70 features
Data Filtering	Our Method 2	Used hyperparameters that returned optimal results in previous experiments
Classification	RFC	Used with 500 estimators and Gini criterion

Table 4.9: Algorithms used for cross temporal validation

Given the present epidemic and general lockdown we are presently experiencing and the existing limitations on computational power, this experiment was run on a computer located in Direção Geral do Território (DGT), whose results are unavailable. Although, preliminary experiments on this topics showed that data filtering algorithms proved ineffective when faced with cross temporal data.

An additional analysis was done with the goal of understanding LULC change over the two most recent LULC maps produced at DGT (2015 and 2018). Figure 4.9 shows a confusion matrix comparing land cover change over these two periods on a sample of approximately one million pixels over the periods of 2015 and 2018.

		COS 2018										
		Territórios artificializados	Agricultura	Pastagens	Superfícies agroflorestais	Florestas	Matos	Espaços descobertos ou com pouca vegetação	Zonas húmidas	Massas de água superficiais	Total	UA
COS 2015	Territórios artificializados	26178	285	223	26	196	170	17	3	69	27167	0.964
	Agricultura	753	269394	7938	667	1362	233	0	1	154	280502	0.960
	Pastagens	364	8339	100905	134	114	432	5	0	91	112616	0.896
	Superfícies agroflorestais	32	790	711	23001	871	28	0	0	41	232483	0.989
	Florestas	224	1421	605	2348	294847	463	216	0	41	300165	0.982
	Matos	58	200	61	10	234	6481	0	2	2	7048	0.920
	Espaços descobertos ou com pouca vegetação	2	12	19	6	563	54	1596	0	0	2252	0.709
	Zonas húmidas	0	1	3	0	5	5	0	565	0	5664	0.998
	Massas de água superficiais	21	140	6	13	214	22	0	0	31682	32098	0.987
	Total	27632	280582	110471	23442	299432	7888	1834	5656	3208	999995	
	PA	0.947	0.960	0.913	0.981	0.985	0.822	0.870	0.999	0.988		0.967

Figure 4.9: LULC change analysis using the COS sampled dataset.

Bibliography

[1] S. Boukir and W. Feng. Identifying and Correcting Mislabeled Satellite Image Data by Iterative Ordering of Ensemble Margins. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 3093–3096. IEEE, jul 2019.

[2] C. E. Brodley and M. A. Friedl. Identifying Mislabeled Training Data. *Journal Of Artificial Intelligence Research*, 11:131–167, jun 1999.

[3] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79, jul 2018.

[4] Annie Farrell, Guiming Wang, Scott A. Rush, James A. Martin, Jerrold L. Belant, Adam B. Butler, and Dave Godwin. Machine learning of large-scale spatial distributions of wild turkeys with high-dimensional environmental data. *Ecology and Evolution*, 9(10):5938–5949, may 2019.

[5] Diego García-Gil, Julián Luengo, Salvador García, and Francisco Herrera. Enabling Smart Data: Noise filtering in Big Data classification. *Information Sciences*, 479:135–152, apr 2019.

[6] Benyamin Ghojogh, Maria N. Samad, Sayema Asif Mashhadi, Tania Kapoor, Wahab Ali, Fakhri Karray, and Mark Crowley. Feature Selection and Feature Extraction in Pattern Analysis: A Literature Review. may 2019.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 770–778. IEEE Computer Society, dec 2016.

[8] Tin Kam Ho. Random decision forests. In *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, volume 1, pages 278–282. IEEE Computer Society, 1995.

[9] Yuan Jiang and Zhi Hua Zhou. Editing training data for kNN classifiers with neural network ensemble. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3173:356–361, 2004.

[10] Reza Khatami, Giorgos Mountrakis, and Stephen V. Stehman. A meta-analysis of remote sensing research on supervised pixel-based land-cover image classification processes: General guidelines for practitioners and future research. *Remote Sensing of Environment*, 177:89–100, may 2016.

[11] Kenji Kira, Larry A Rendell, et al. The feature selection problem: Traditional methods and a new algorithm. In *Aaai*, volume 2, pages 129–134, 1992.

[12] Igor Kononenko, Edvard Šimec, and Marko Robnik-Šikonja. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, 7(1):39–55, 1997.

[13] Weijia Li, Haohuan Fu, Le Yu, Peng Gong, Duole Feng, Congcong Li, and Nicholas Clinton. Stacked Autoencoder-based deep learning for remote-sensing image classification: a case study of African land-cover mapping. *International Journal of Remote Sensing*, 37(23):5632–5646, dec 2016.

[14] Fei Tony Liu, Kai Ming Ting, and Zhi Hua Zhou. Isolation forest. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 413–422, 2008.

[15] (ESTAT) E4. LUCAS. Lucas 2015 (land use / cover area frame survey). Technical reference document C1. Instructions for Surveyors:140, 2015.

[16] Masoud Mahdianpari, Bahram Salehi, Mohammad Rezaee, Fariba Mohammadimanesh, and Yun Zhang. Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery. *Remote Sensing*, 10(7):1119, jul 2018.

[17] Aaron E. Maxwell, Timothy A. Warner, and Fang Fang. Implementation of machine-learning classification in remote sensing: An applied review. *International Journal of Remote Sensing*, 39(9):2784–2817, 2018.

[18] Claudia Paris, Lorenzo Bruzzone, and Diego Fernandez-Prieto. A Novel Approach to the Unsupervised Update of Land-Cover Maps by Classification of Time Series of Multispectral Images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(7):4259–4277, 2019.

[19] Charlotte Pelletier, Silvia Valero, Jordi Inglada, Nicolas Champion, Claire Marais Sicre, and Gérard Dedieu. Effect of Training Class Label Noise on Classification Performances for Land Cover Mapping with Satellite Image Time Series. *Remote Sensing*, 9(2):173, feb 2017.

[20] Charlotte Pelletier, Silvia Valero, Jordi Inglada, Gérard Dedieu, and Nicolas Champion. Filtering mislabeled data for improving time series classification. In *2017 9th International Workshop on the Analysis of Multitemporal Remote Sensing Images, MultiTemp 2017*. Institute of Electrical and Electronics Engineers Inc., sep 2017.

[21] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep Learning is Robust to Massive Label Noise. may 2017.

[22] Swalpa Kumar Roy, Gopal Krishna, Shiv Ram Dubey, and Bidyut B. Chaudhuri. HybridSN: Exploring 3-D-2-D CNN Feature Hierarchy for Hyperspectral Image Classification. *IEEE Geoscience and Remote Sensing Letters*, pages 1–5, jun 2019.

[23] Weiwei Yuan, Donghai Guan, Qi Zhu, and Tinghuai Ma. Novel mislabeled training data detection algorithm. *Neural Computing and Applications*, 29(10):673–683, may 2018.

[24] Weining Zhang, Dong Wang, and Xiaoyang Tan. Robust Class-Specific Autoencoder for Data Cleaning and Classification in the Presence of Label Noise. *Neural Processing Letters*, 50(2):1845–1860, 2019.

[25] Xiaodong Zhang, Guanzhou Chen, Wenbo Wang, Qing Wang, and Fan Dai. Object-Based Land-Cover Supervised Classification for Very-High-Resolution UAV Images Using Stacked Denoising Autoencoders. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(7):3373–3385, jul 2017.

[26] Xinbin Zhang. An Improved Method of Identifying Mislabeled Data and the Mislabeled Data in MNIST and CIFAR-10. *SSRN Electronic Journal*, apr 2018.