Evaluation of machine learning methods for pixel-level cloud detection considering inter-location validation at different granularities

1 Introduction

The Copernicus Sentinel-2 mission offers optimized bands for this task and provides unprecedented amounts of data in terms of spatial sampling, global coverage, spectral coverage and repetition rate. Detecting clouds and their shadows, cirrus clouds, snow, shadows, and areas of clear sky - is a critical step in pre-processing images from optical satellites images. The difficulty lies in the great diversity of cloud types and land surface landscapes. It is already common to confuse bright landscapes with clouds, and it is difficult to detect semi-transparent clouds (a mixture of cloud and ground signals).

In 2016, Hollstein et al [1] have built a dataset consisting of approximately 6.4 million hand-labeled pixels from 108 Sentinel-2 scenes, sampled roughly uniformly around the world and throughout the year to ensure complete climate coverage. This dataset is particularly well suited for testing spatial transfer methods, i.e., we test machine learning models on data from a location that is not involved in the training phase. Here, we evaluate the methods by defining locations at different levels of granularity (or geographical scales) : *tiles*, *countries* and *continents*.

The objective of this study is to evaluate common machine and deep learning algorithms for the classification of the pixels from multispectral images to understand the behavior of the model when it attempts to generalize to a new location that is more or less far from the training pixels depending on the level of granularity.

2 Data

The pixels in the Sentinel-2 image database cover the natural variability present in satellite image observations, due to the diversity of landscapes, climates and illumination properties. Each spectrum included has one of the following labels: cloud, cirrus, snow/ice, shadow, water and clear sky. Instead of considering only the cloud/non-cloud task, we work on all the classes available for the detection task, to better understand the confusion we may have between them. We also considered all available bands as inputs for the models, which means that we will get the pixel detector at a resolution of 60 m, since we take the bands that have this resolution.

In Figure 1, we have computed the average spectral profile to analyze the similarities and differences between the classes. The axis represents the center of the band wavelength. The ordinate represents the average reflectance value per class and the confidence interval calculated from the standard deviation between the different tiles. We can see that on average, the classes are well separable in terms of average spectrum.

In the Tables 1 and 2, we summarize respectively the number of countries, tiles and pixels per continent, and the percentage of pixels of each class for a given continent.

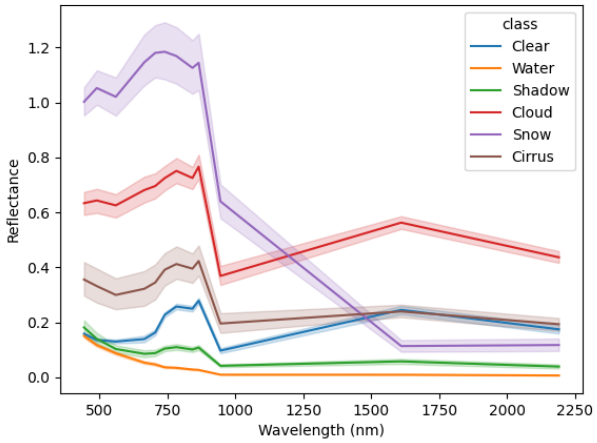

Figure 1: Spectrum profile per class

Continent	#Countries	#Tiles	#Pixels
Africa	10	24	1,468,277
America	5	23	1,385,328
Asia	3	13	866,103
Europe	9	23	1,174,205
Oceania	4	14	713,529
Total	31	97	5,607,442

Table 1: Number of countries, tiles and pixels grouped by continent in the dataset. Oceania and Asia are less represented than Europe, America and Africa.

Class	Africa	America	Asia	Europe	Oceania
Clear	0.252	0.201	0.258	0.156	0.230
Water	0.129	0.223	0.127	0.207	0.178
Snow	0.168	0.105	0.207	0.200	0.224
Cirrus	0.117	0.071	0.199	0.166	0.094
Cloud	0.301	0.081	0.045	0.094	0.252
Shadow	0.032	0.318	0.164	0.177	0.022

Table 2: Percentage of pixels represented by class and continent. Africa and Oceania have a lot of cloudy pixels, while America has more than 30% of pixels identified as shaded. This imbalance may affect the performance of the model when we try to predict all the pixels of a test continent.

3 Experiments

3.1 Multi-class classification models

For the pixel multiclass detection task, one ensemble learning methods are evaluated that have already proven successful in the field of classification of multispectral images [2] and cloud detection [3], namely Light Gradient-Boosting Machine (LGBM). The LightGBM estimator [4] was chosen for classification to be at least partially available to compare results with the example in the work of Anze Zupanc [1]. LightGBM is an efficient algorithm based on the gradient boosting decision tree machine learning method. We used a One-vs-the-rest (OvR) approach for our multiclass classification approach, in which a binary classifier is trained for each class C. Data from class C are treated as positive, and all other data as negative. The OvR approach used was developed by Halford [2].

In addition, we attempt to analyze multispectral data from a sequential point of view [5], i.e. each multispectral sample (pixel) is seen as a data sequence. CNN models make the most of the spatial structure of the data by applying convolutions in both x and y dimensions. One of the application of CNNs in remote sensing remains the classification of hyperspectral images, where 1D-CNNs across the spectral dimension [6] have been successfully tested. A summary of the chosen architecture is detailed in Table 3. Each fully connected and convolutional hidden layer is followed by a batch normalization and a ReLU (Rectified Linear Unit) activation function. The deep learning models were trained for 100 epochs with a batch size of 32 and a learning rate of 10^{-4} using the Adam optimizer.

Block	Layer description
1	Conv(nf=32, k=3, s=1, drop=0.8)
2	Conv(nf=64, k=3, s=2, drop=0.8)
3	Conv(nf=128, k=3, s=2, drop=0.8)
4	Global Avergage Pooling()
5	FC(n=32, drop=0.5)

Table 3: Details of the CNN architecture: Conv stands for Convolution operation, nf is the number of filters, k is the kernel size, s is the stride value. Padding is set to "SAME". The stride value is set to 2 for the second and third blocks to help eliminate unnecessary computational overhead and to reduce the size of the features, where we likely have redundant information between regions of the spectrum.

We also considered Focal loss [7]. It was originally proposed to solve imbalance problems that occur in object detection model training. For binary classification problem with p_t is a function of the true labels, it is defined as follows:

[1] https://medium.com/sentinel-hub/improving-cloud-detection-with-machine-learning-c09dc5d7cf13
[2] https://maxhalford.github.io/blog/lightgbm-focal-loss/first-order-derivative

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma} log(pt) \quad (1)$$

The weighting factor α_t balances the modulating factor. With $\gamma = 0$, the Focal loss is equivalent to the cross-entropy loss. To summarize, Focal loss can be interpreted as a binary cross-entropy function multiplied by a modulating factor $(1-p_t)$ which reduces the contribution of easy-to-classify samples. We set $\gamma = 2$ and $\alpha = 0.25$ as suggested in [7].

3.2 Validation approaches

3.2.1 Machine learning methods

We consider here the use case where we wish to detect the pixel class for a new location, and thus avoid the unrealistic case where the data from the geographic test location is available for model training. Therefore, the model hyperparameters were selected using the leave-one-group-out method, with the groups divided by geographical location. It aims to produce validation and test sets must be representative of the new unseen data.

For example, if we consider the granularity level *continents* of our dataset that covers the five continents, it means that we consider four continents for model training and one for testing. Among these four training continents, one is used for validation and the other three for model training. We repeat this on each continent dedicated to the training of the model, i.e. here four times. Then, once the model hyperparameters are selected, we train the model on all four training continents and the prediction for each pixel in the test continent is compared to the associated labeled pixel. This tests the capability of the model in a real-world scenario in which we force the model to generalize well to a different location using only data collected from other different locations.

For the granularity levels *countries* and *tiles*, we want the number of observations in the training, validation and test sets to be aproximately similar to the granularity level *continents*. By analogy, we separate the total number of countries (resp. tiles) into 5 disjoint sets of equal proportion, such that 4 will be dedicated to learning and one to testing. Then, we iterate on the 4 validation sets to find the hyperparameters for a given test set. Finally, we iterate over each test set to obtain a measure of the error dispersion.

A summary of the validation approach to train the machine learning methods is despicted in Figure 2.

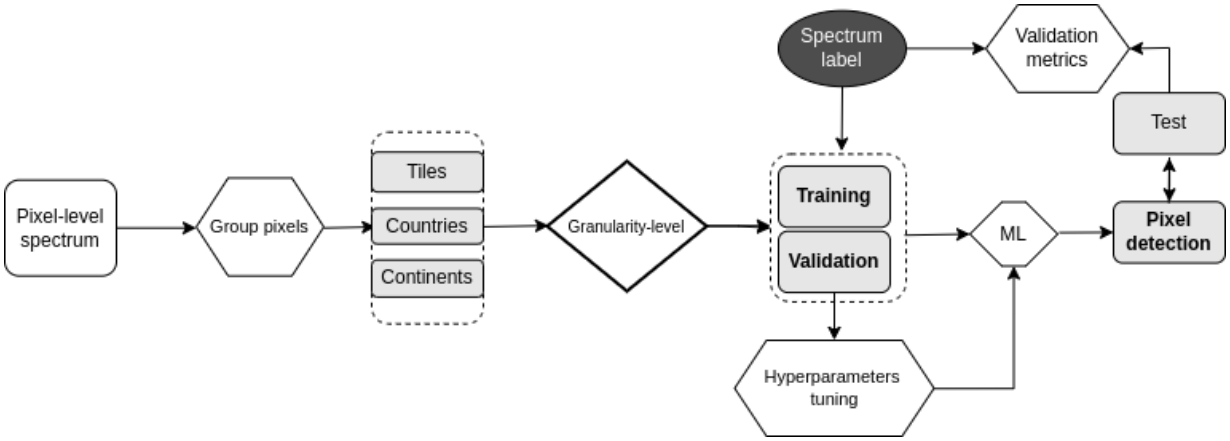

Figure 2: Setting hyperparameters for the machine learning models, depending on the level of granularity of the validation sets (*tiles*, *countries* or *continents*). Then, once the parameters are set, we train the algorithms on the training data set, which thus includes the validation set.

The hyperparameters are selected with respect to the best accuracy score obtained in average on the validation folds. Due to limited computational resources, we randomly sample pixels without replacement in the training and validation sets. We set the probability to 0.1, with a fixed random seed so that the data subset is the same for each method.

3.2.2 Deep learning methods

For deep learning models, we use the validation set to save weights when the model has the best performance metric on it. At the end, we select the model with the best score among the validation sets for a given test set. Therefore, deep learning models have less input data than machine learning models, as these last ones train on the entire dataset when hyperparameters are found. One idea not implemented in this study would be to create a

set of deep learning methods such that a model represents a validation set on which it has been trained. At the end, we could compute a weighted average of predictions on the test data based on the accuracy obtained on the validation set. The goal is to reduce the random uncertainty caused by the random initialization of deep learning methods, which can cause arbitrary errors on test samples that are not in the distribution of the training data [8].

4 Results and discussion

In this section, we will compare the results of the machine learning methods presented in section 3.1, according to the different levels of granularity for spatial transfer *tiles*, *countries*, and *continents*. We compare the results in terms of F1-score (or F-measure) that combines precision and recall into a single metric. Aggregation by class is performed by a weighted average, so that each class' contribution to the F1 average is weighted by its size.

Method	Level *tiles*	Level *countries*	Level *continents*
LGBM	**0.882 ± 0.04**	0.853 ±0.07	0.899 ±0.04
CNN	0.789 ±0.07	0.844 ±0.05	0.866 ±0.06
LGBM_{FOCAL}	0.881 ±0.04	**0.854 ± 0.07**	**0.899 ± 0.04**
CNN_{FOCAL}	0.799 ±0.09	0.799 ±0.07	0.839 ±0.06

Table 4: Average F1-score per type of granularity level to assess the spatial transfer (*tiles*, *countries* or *continents*)

In Table 4, we can first see that the LGBM outperform CNN models in average perform in terms of weighted F1 measure, which suggests that the convolution on the spectrum dimension. Moreover, to our surprise, a country-level validation yields worse results than a continent-level one, and CNNs were less successful in validating by tile than by country or continent. However, the CNN would require additional tuning of the hyperparameters since we chose an architecture that seemed appropriate at random. In Table 5, we have reported the average F1-score per ontinent for each method. Although CNN performed less well on average, it proved to be the best method when it came to the Asian continent with an F1-score of 0.878 against 0.826 for LGBM_{FOCAL}. Finally, in both tables, we can see that the Focal fonctino has degraded the CNN scores on average, and gives very similar results for the LGBM. However, in Figures 3, 4 and 5 , we will take a closer look at the confusion matrices to see if we have differences in model behavior between these two losses.

Method	Africa	America	Asia	Europe	Oceania
LGBM	0.902	0.956	0.816	**0.898**	0.**918**
CNN	0.894	0.890	**0.878**	0.755	0.914
LGBM_{FOCAL}	**0.906**	**0.957**	0.826	0.894	0.915
CNN_{FOCAL}	0.904	0.877	0.806	0.746	0.861

Table 5: Average F1 score per continent for each method

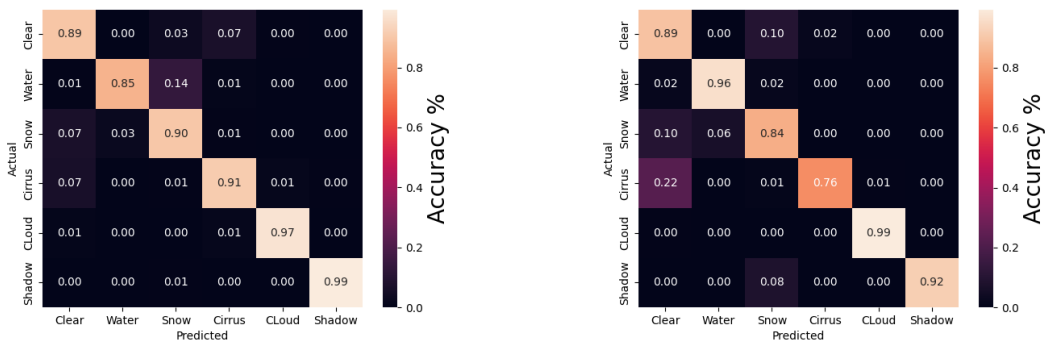

Figure 3: Comparison of the levels *tiles* and *countries* in term of confusion matrices for LGBM_{FOCAL}

In the Figure 3, we compare the confusion matrices for LGBM_{FOCAL} at the *tiles* and *countries* level. While the level *tiles* shows little confusion between classes, the country level shows that the model only classifies with 76% accuracy the *Cirrus* class and it confused it with the *Clear* class. We can thus deduce that a tile-based validation approach inflates the evaluation of the method, given a possible spatial autocorrelation between tiles that are close

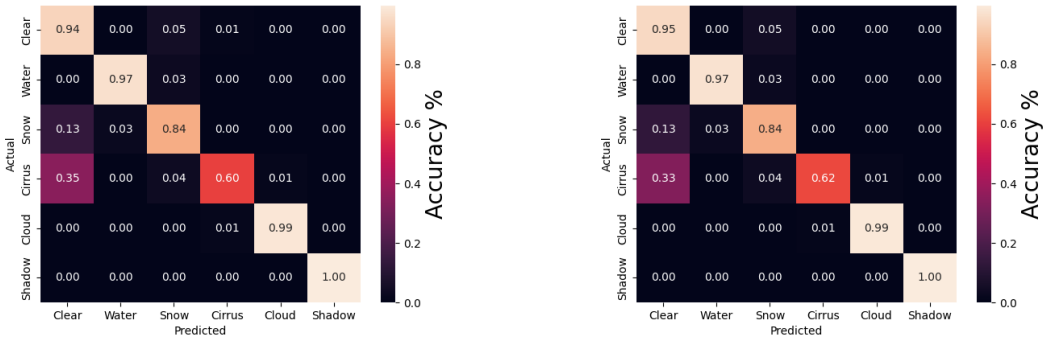

Figure 4: Comparison between cross-entropy and focal losses using LGBM method for the level *continents*, where all the pixels are from Africa

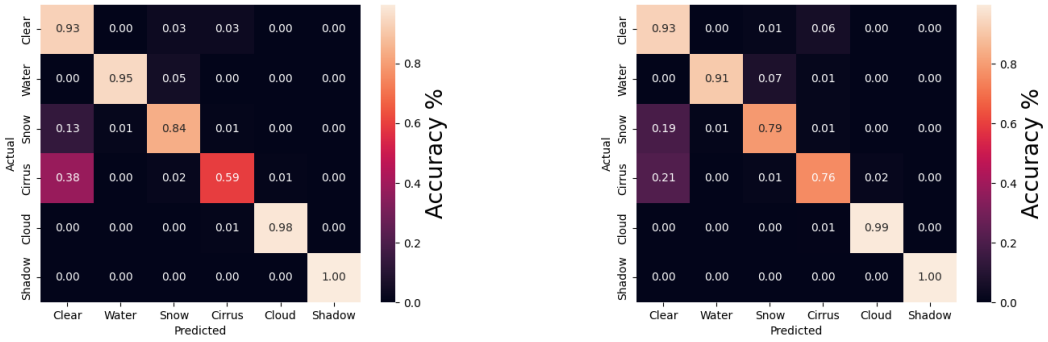

Figure 5: Comparison between cross-entropy and focal losses using CNN method for the level *continents*, where all the pixels are from Africa

to each other and that are in both the training and test set. In the Figures 4 and 5, we plot the confusion matrices at the level of LGBM and CNN respectively, where we compare the accuracy per class as a function of the loss function used to optimize the model (Cross-Entropy or Focal). Although the different in term of F1 score in Table 3 was not significant, we can see that the major improvement of using Focal loss was to detect the class *Cirrus*. For LGBM (resp. CNN), we can predict 62% (resp. 76%) of the pixels considered as Cirrus using the Focal loss, compared to 60% (resp. 59%) using the standard Cross-Entropy loss for this given test set.

5 Conclusion

This study analyzed and compared the performance of methods based on data for cloud detection using Sentinel-2. A grouping at several geographical scales was carried out to analyze the ability of the model to classify pixels for a new geographical location. More particularly, we were interested in classifying pixels that are geographically distant between the training and test dataset, which can cause a distribution shift. We evaluated the spatial transfer of the CNN1D and LightGBM methods at different geographical scales (*tiles*, *countries* or *continents*), such that the locations of the training samples are not represented in the test samples. To our surprise, the best scores were found when we sought to predict at the level *continents*, and the worst at the level *countries*. The most difficult class to predict for the latter is Cirrus, but we went for example from about 60% to 75% accuracy in Africa using a convolutional neural network with a Focal loss function.

Our next work will focus on the study of domain adaptation methods, where the goal is to align the distribution of the source and target domains. We hope that this can further improve accuracy when we attempt to predict all pixels in a given continent. The code to reproduce the data preprocessing and experiments has been shared on the following github link `https://github.com/j-desloires/cloud-mask`.

References

[1] A. Hollstein, K. Segl, L. Guanter, M. Brell, and M. Enesco, "Ready-to-use methods for the detection of clouds, cirrus, snow, shadow, water and clear sky pixels in sentinel-2 msi images," *Remote Sensing*, vol. 8, no. 8, p. 666, 2016.

[2] C Candido, A. Blanco, J Medina, *et al.*, "Improving the consistency of multi-temporal land cover mapping of laguna lake watershed using light gradient boosting machine (lightgbm) approach, change detection analysis, and markov chain," *Remote Sensing Applications: Society and Environment*, vol. 23, p. 100 565, 2021.

[3] T. Řezník, J. Chytrỳ, and K. Trojanová, "Machine learning-based processing proof-of-concept pipeline for semi-automatic sentinel-2 imagery download, cloudiness filtering, classifications, and updates of open land use/land cover datasets," *ISPRS International Journal of Geo-Information*, vol. 10, no. 2, p. 102, 2021.

[4] G. Ke, Q. Meng, T. Finley, *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[5] H. Wu and S. Prasad, "Convolutional recurrent neural networks forhyperspectral data classification," *Remote Sensing*, vol. 9, no. 3, 2017, ISSN: 2072-4292. DOI: 10.3390/rs9030298. [Online]. Available: https://www.mdpi.com/2072-4292/9/3/298.

[6] F. Hu, G.-S. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14 680–14 707, 2015.

[7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[8] N. Lang, N. Kalischek, J. Armston, R Dubayah, and J. Wegner, "Global canopy height regression and uncertainty estimation from gedi lidar waveforms with deep ensembles," *Remote Sensing of Environment*, vol. 268, p. 112 760, Jan. 2022. DOI: 10.1016/j.rse.2021.112760.