Incremental constrained clustering with application to remote sensing images time series

Baptiste Lafabregue[1], Pierre Gançarski[1], Jonathan Weber[2], Germain Forestier[2]

[1] ICube, Université de Strasbourg, France, {*lafabregue, gancarski*}*@unistra.fr*

[2] IRIMAS, Université de Haute Alsace, France, {*jonathan.weber, germain.forestier*}*@uha.fr*

***Abstract*—Automatically extracting knowledge from various datasets is a valuable task to help experts explore new types of data and save time on annotations. This is especially required for new topics such as emergency management or environmental monitoring. Traditional unsupervised methods often tend to not fulfill experts' intuitions or non-formalized knowledge. On the other hand, supervised methods tend to require a lot of knowledge to be efficient. Constrained clustering, a form of semi-supervised methods, mitigates these two effects, as it allows experts to inject their knowledge into the clustering process. However, constraints often have a poor effect on the result because it is hard for experts to give both informative and coherent constraints. Based on the idea that it is easier to criticize than to construct, this article presents a new method, I-SAMARAH, an incremental constrained clustering method. Through an iterative process, it alternates between a $clustering$ phase where constraints are incorporated, and a $criticize$ phase where the expert can give feedback on the clustering. We demonstrate experimentally the efficiency of our method on remote sensing image time series. We compare it to other constrained clustering methods in terms of result quality and to supervised methods in terms of number of annotations.**

***Index Terms*—Constrained clustering, User feedback, Time series, Remote sensing**

I. Introduction

Following the growth of data production, machine learning has seen an increasing interest these last years, leading to the production of robust and effective methods. These methods often rely on a lot of annotated data to perform well, however, in most cases such annotations are not easily available. New application domains, like time series and more specifically in remote sensing, are particularly prone to this issue. On one hand, due to the novelty of such data, the semantic fields (thematic classes, nomenclatures...) appropriate to qualify the data may not be perfectly defined. This tends to make the creation of learning datasets difficult, if not impossible. On the other hand, even if formalized knowledge exists, manual annotation remain fastidious and time-consuming.

In such a situation, where a lot of data is available but not labeled, unsupervised clustering approaches have been shown to be relevant and very effective [1]. Nevertheless, experts often have intuitions or partial knowledge of the data (e.g. idea of existing thematic classes, labels for some objects, . . .). Unfortunately, they do not generally have efficient mechanisms to provide this background knowledge directly to the clustering process. So, constrained clustering, a form of semi-supervised clustering, has been proposed to circumvent this problem [2].

In this kind of approach, experts express their background knowledge in the form of constraints that the clustering should respect, constraints that are easier to define than labels. However, most of the time these methods require a large amount of constraints to be significantly effective [3], [4]. Thus, experts face a similar problem as in supervised scenarios: how to give the set of relevant annotations as small as possible?

The article [2] highlights two limitations of constraints use. First, it shows that most constraints given at random will be satisfied whether they are fed to the clustering method or not as they correspond to a strong intrinsic reality. Secondly, it also demonstrates that constraints' quality is an important factor that can help reduce the number of required constraints.

To tackle these issues, an extension of constrained clustering methods, called *active constrained clustering*, has been recently proposed. It is based on the idea that knowledge can be interactively discovered during the clustering process itself. It consists in submitting chosen queries to the expert to allow the system to create constraints more relevant and more exploitable by the constrained clustering method [5], [6]. Experiments show that these methods tend to dramatically reduce the number of required constraints. Unfortunately, the system's queries can involve objects out of the experts' domain of interest or out of their expertise domain.

Therefore, the aim of our work is to define a method able to guide experts in defining constraints that are both likely to be useful and that allow them to focus on the part of the data that they are interested in. Concretely, this article proposes an innovative method that aims to answer to these two points, called Incremental SAMARAH (I-SAMARAH) and based on SAMARAH method [7]. It differs from existing approaches by changing the way constraints are given by the user. In this method, the user can provide additional constraints on the fly based on the algorithm's results and thus, thanks to those, guide the progress of the analysis. Indeed, it seems simpler for an expert to propose new constraints based on a result rather than from scratch, i.e. only based on the raw data. The paper also validates this method on remote sensing time series analysis. Indeed, the remote sensing domain presents the majority of the issues introduced above (lack of knowledge, high number of potential thematic classes, . . .).

This work was supported by the ANR HERELLES project (grant ANR-20-CE23-0022) and the ANR HIATUS project (grant ANR-18-CE23-0025) and the ANR TIMES project (ANR-17- CE23-0015) of the French Agence Nationale de la Recherche

Indeed, even though studies were already conducted on satellite image time series in the past decades [8], [9], they were limited by three major factors: low data resolution, data sparsity, and acquisition cost. These limitations have been strongly mitigated with various initiatives, such as the Copernicus[1] European Union's program (e.g. Sentinel-2 satellites launched in 2015-2017).This new paradigm offers many perspectives to monitor or analyze land coverage evolution [8].

In the rest of this article, we first discuss related work on (active) constrained clustering in Section II. In Section III we present our new method I-SAMARAH. In Section IV we present the material and methodological choices made for the experiments. Then, in Section V we present and discuss our test results. We show that our method outperforms other constraints clustering and active constrained clustering methods, even with a fewer number of constraints. Finally, in Section VI, we present our conclusions and discuss the perspectives of our work.

II. Background and Related Work

A. Pairwise constraints and clustering

Clustering is an essential tool in data analysis but its results will often depend on its optimization function and its initialization state. Hence, it may often fall in local optima that will not match the user requirements or expectations. Constrained clustering aims to incorporate knowledge in the clustering process to regulate its result. The user expresses this knowledge through a set of constraints. Constraints can have different forms and can be distinguished between the one related to clusters and the one related to instances [10].

We distinguish two main types of constraints. First, the ones on the cluster-level, e.g. the number of clusters K or their maximum diameter [2]. Second, the ones on the instance-level, with mainly the pairwise constraints that are of two kinds, *must-link* (ML) and *cannot-link* (CL). ML constraints indicate that two instances should be assigned to the same cluster and CL constraints that they should be assigned to different ones. This type of constraint has been extensively used, which makes it easier for method comparison. These constraints are also intuitive for the user and easy to give, as they just express if two objects (e.g. pixels or segments in remote sensing) should be grouped together or not. In consequence, this article will only focus on pairwise constraints.

Many methods use pairwise constraints, introduced first by Wagstaff et al. with the COP-KMEANS method [11]. COP-KMEANS is an extension of the widely used K-means clustering method. Most of the proposed constrained clustering methods follow the same scheme and extend a standard clustering method, including K-Means [11], [12], Spectral Clustering [4], SOM. Constraints may be used at different levels of the clustering process. Some methods use them to update the similarity of the constrained instances [4], to control the clustering assignment [11], [13], or as a simple guidance [7]. However, they either need a very high number of annotations, or a large portion of the data to be annotated for small datasets.

[1]https://www.copernicus.eu/fr

In all these methods, the constraints quality plays an important role in constrained clustering performance and is an important track to reduce the number of required constraints. [14] Davidson et al. [14] have shown that constraints are effective if they fulfill two criteria:

- to be informative: constraints should not be satisfied by the algorithm on its own (i.e. unconstrained clustering).
- to be coherent: constraints should not contradict themselves. Davidson et al introduced coherence as a measure that quantifies "the amount of agreement between the constraints themselves, given a metric that specifies the distance between points" [14].

To this end, active constrained clustering methods were proposed to ensure the constraints' quality.

B. Active constrained clustering and user feedback

Different strategies are proposed to select constraints in the literature [5]. Some methods focus on finding constraints that bring the most information or with the most uncertainty [5], but others try to find the ones that will have the larger effect on the clustering [6], [15] or reduce the potential errors of the clustering.

Another approach, inspired by active learning [5], has been proposed by Cohn et al. [16]. This method relies on user feedback to generate constraints. In this approach, the user starts from an initial clustering. Then, the user gives indications of his/her agreement or disagreement with the proposed result in the form of pairwise constraints between clustered instances. Constraints are integrated into the similarity measure, the KL divergence to the mean. This measure was proposed for document classification, however, to the best of our knowledge, no adaptation to temporal data has been done. Davidson et al. studied another approach to circumvent the problem, based on the idea that it is often more efficient to update an existing clustering to satisfy new (and old) constraints rather than re-clustering the entire dataset from scratch [17]. These two articles confirm that user feedback is more effective than randomly selected constraints. They also argue that the number of constraints does not need to be high, and even more, that if this number increases too much, the results can deteriorate.

With I-SAMARAH, we propose a new approach that extends the collaborative clustering framework proposed by SAMARAH [7] by adding a mechanic that allows the expert to give new constraints in an iterative way as described in the next section. I-SAMARAH aims to take advantage of both good results obtained by collaborative methods [2] and iterative user feedback described above.

III. Incremental Collaborative Clustering with Samarah

A. Collaborative clustering

Collaborative clustering is similar to ensemble clustering which considers that the information offered by different sources and different methods are complementary. It consists in making multiple clustering methods (agents) collaborate to reach an agreement on one data partitioning.

The SAMARAH [7] main concept is to make different clustering algorithms (agents) collaborate and modify their results until they reach a strong similarity. The algorithm is divided in three steps:

- Initialization: each agent computes its clustering
- Collaboration: mutual refinement of agents' clustering results
- Unification: combination of agents' clustering results

In the first step, every agent applies its clustering method to the data (lines 2 to 3 of Algorithm 1). Each agent may use a different algorithm or the same but with different parameter values (e.g. hyperparameters, initialization, a data's subset).

In the second step, results computed previously are refined to maximize an optimization function, called collaborative criterion (line 4 to 14 of Algorithm 1). During the refinement stage, each result is compared with the set of results proposed by the other methods. It evaluates the dissimilarity between the different agent's results in order to define a set of differences, also named conflicts. A conflict is defined as a non-similarity between a cluster from an agent and its most overlapping cluster in another agent's cluster. Once these conflicts are identified, the objective is to modify the results to reduce these differences, i.e. resolving the conflicts [18]. Conflicts are sorted from higher to lower dissimilarity. Then, conflicts are resolved one after the other by either merging clusters, splitting clusters, or re-clustering clusters (delete the cluster) iteratively. This step can be seen as questioning each result according to information provided by the other actors in the collaboration. For each conflict, different combinations are proposed, and the one that maximizes the local collaborative criterion is retained. The local criterion between two agents i and j can be defined as follow:

$$\gamma^{(i,j)} = \frac{1}{2}(p_s.(\frac{1}{n_i}\sum_{k=1}^{n_i}\omega_k^{(i,j)} + \frac{1}{n_j}\sum_{k=1}^{n_j}\omega_k^{(j,i)}) + p_q.(\delta^i + \delta^j))$$
$$\text{, where } p_s + p_q = 1 \tag{1}$$

Where $\omega_k^{(j,i)}$ is the similarity between the cluster k of agent i's clustering and its most overlapping cluster in agent j's clustering, δ_i the internal quality of agent i's clustering (e.g. the clusters' compactness), and p_s, p_q are weights that allow the user to balance between similarity and internal quality. After the best combination selection, we evaluate its impact on the global clustering. The global criterion can be defined as a sum of local criterion between each m agents:

$$\Gamma = \frac{1}{m}\sum_{i=1}^{m}\Gamma^i, \quad \text{where} \quad \Gamma^i = \frac{1}{m-1}\sum_{j=1, j\neq i}^{m}\gamma^{(i,j)} \tag{2}$$

More details on similarity computation of conflict detection and resolution can be found in [18]. After multiple refinement iterations, the agent's results are expected to be more similar than before the collaboration started. This step can be seen as an exploratory process, guided by the global criterion.

During the third and final step, the refined results are combined to propose a final and unique result, which is simplified due to the similarity of agents' results (line 15 of Algorithm 1).

Algorithm 1 SAMARAH

```
1:  procedure SAMARAH(dataset $\mathcal{O}$, set of clust. agents $\mathcal{A}$)
2:      for each agent $a_i$ in $\mathcal{A}$ do
3:          Compute clustering of $\mathcal{O}$ with method $a_i$
4:      Create the set of all conflicts $\mathcal{C}$, by evaluating the
        dissimilarities between pairs of results
5:      Let $\mathcal{E} = \Gamma$ be the evaluation of the initial results
        according to the collaborative criterion
6:      while $\mathcal{C}$ not empty do
7:          Choose a conflict $c$ to solve from $\mathcal{C}$
8:          Local resolution of the $c$ with the involved agents
9:          Let $\mathcal{E}' = \Gamma$ be the evaluation of the updated results
        according to the collaborative criterion
10:         if $\mathcal{E}' > \mathcal{E}$ then
11:             $\mathcal{E} = \mathcal{E}'$
12:             Apply modifications to the learning agents
13:             Compute the new set of conflicts and update $\mathcal{C}$
14:     Compute the final result from the agents with an
        adapted voting algorithm
```

B. Integration of constraints

Forestier et al. proposed an adaptation of SAMARAH to integrate constraints in the process [7]. Background knowledge is incorporated into the collaborative criterion. The criterion is extended to ensure that constraints are represented when resolving conflicts. This way a wide range of background knowledge can be integrated.

It requires a function to be defined that measures the satisfaction of the prior knowledge in agent n's solution ($\mathcal{R}^n$) on the range $[0, 1]$. In our implementation that support *must-link* (ML) and *cannot-link* (CL) constraints, the rate of satisfied constraints by the agent n, θ_n, is used. The local criterion in SAMARAH algorithm is then modified as follows:

$$\begin{aligned}\gamma^{(i,j)} =& \frac{1}{2}(p_s.(\frac{1}{n_i}\sum_{k=1}^{n_i}\omega_k^{(i,j)} + \frac{1}{n_j}\sum_{k=1}^{n_j}\omega_k^{(j,i)}) + p_q.(\delta^i + \delta^j) \\ &+ p_c.(\theta^i + \theta^j)), \text{ where } p_s + p_q + p_c = 1\end{aligned} \tag{3}$$

This modification causes a balance between the background knowledge and the distance metric to be pursued during conflict resolution. Therefore, constraints are used as a guideline for the clustering process, so constraints are not ensured to be satisfied. This might be considered a disadvantage if constraints must be fulfilled, but it also relaxes the problem of over-constraining that usually limit the constraints' effect [2].

C. User feedback

When users are queried feedback, they want that the new clustering takes into account the new information provided

(constraints, labels, ...), but also that the new clustering is as similar as possible to the previous clustering. This is required because users have to find their bearings between two incremental steps. The integration of user feedback supposes the possibility to launch the clustering process iteratively to take user constraints into account at each iteration. To do so, we let a first unsupervised SAMARAH run produce an initial clustering result. After the user gives its feedback, a new run of constrained SAMARAH is launched, but agents keep their current state (e.g. centroïds value). However, as SAMARAH aims to reduce dissimilarity among agents, the final agents' results should be strongly similar. Thus, it tends to reduce the number of conflicts and therefore the exploration space of the collaboration.
To solve this problem, we decided to force artificially the dissimilarity between agents' results. To this end, we spread the constraints among the set of agents to add clusters in each agent's current clustering.

This operation is at the moment only implemented for centroid-based methods. In this implementation, we add centroids for each constraint, as described in algorithm 2. For an ML constraint, we add a new centroid that is the average of the two constraint's instances and for a CL constraint, we add the two involved instances as new centroids, see figure 1. For other methods, a study is ongoing. Note that adding new clusters is not a problem as SAMARAH, by construction, will remove (by merging or re-clustering) unnecessary clusters. The final method is summarized in figure 2.

Algorithm 2 Generate dissimilarity between agents

```
1: procedure GENERATE_DISSIMILARITY(set of learning agents $\mathcal{A}$, set of constraints C $\subseteq \mathcal{O} \times \mathcal{O}$)
2:     for each constraint $c$, between $c1$ and $c2$, in $C$ do
3:         $A' = \emptyset$
4:         for each agent $a_i$ in $\mathcal{A}$ do
5:             if $a_i$ does not satisfy $c$ then
6:                 $A' = A' \cup a_i$
7:         if $A' \neq \emptyset$ then
8:             Pick at random $a$ from $A'$ with $Centers^a$ its set of centroïds,
9:             if $c$ is a ML constraints then
10:                $Centers^a = Centers^a \cup \{avg(\{c1; c2\})\}$
11:            if $c$ is a CL constraints then
12:                $Centers^a = Centers^a \cup \{c1; c2\}$
```

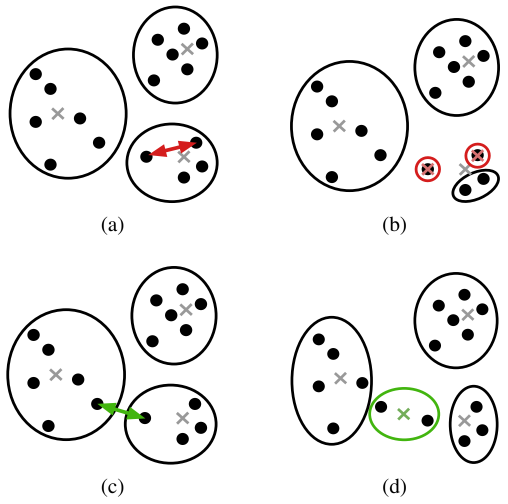


Fig. 1. Illustration of dissimilarity generation when a CL constraint (a) is used and the result afterwards (b), respectively (c) and (d) with a ML constraint. Crosses are representing clusters' centroids

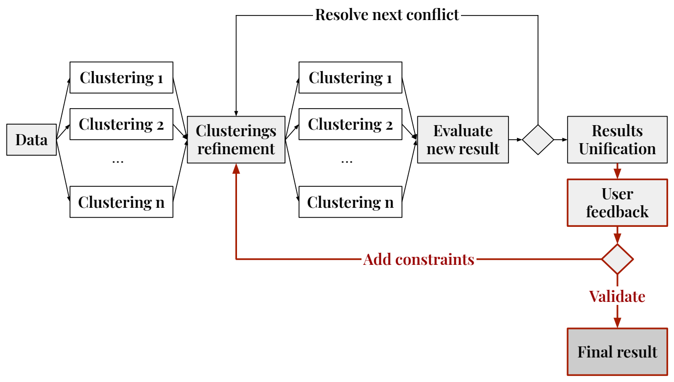


Fig. 2. I-SAMARAH method

IV. MATERIALS

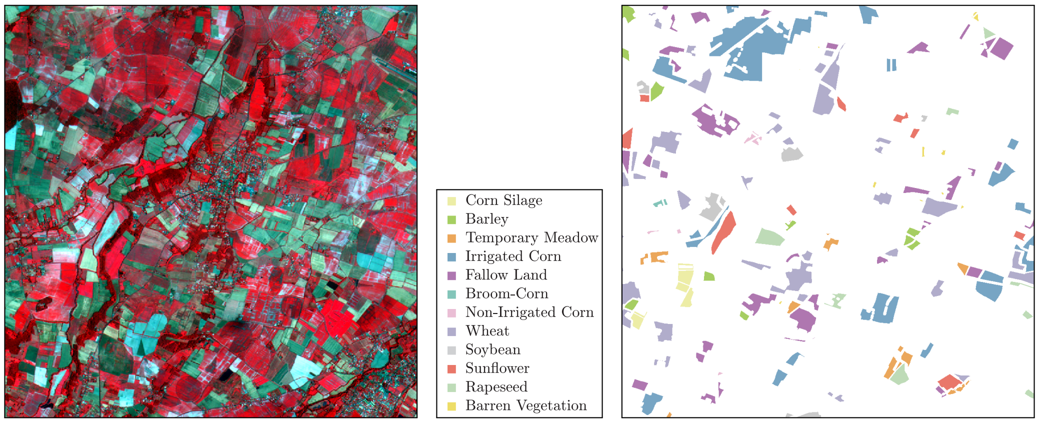
As presented in the introduction, time series in remote sensing is a crucial problem where the experts have often a partial knowledge of the data [1]. Because of that, they are looking for methods to support them in their research that require as few annotations as possible. Furthermore, some visualization tools are already available to help experts analyze remote sensing clustering results. Thus, validating our method on this kind of data seems more beneficial for the experts. We used the FODOMUST framework [19] in our experiments.

We first present the data used in section IV-A. In section IV-B we present methods used for the comparison. In section IV-C we explain how user feedback is provided. Then, in section IV-D we present the metric and the protocol used for the evaluation. Finally, in section IV-E we present the reference data.

A. Remote sensing time series

To evaluate our method and facilitate the comparison between existing constrained clustering methods, we choose two datasets used in a previous review on constraint clustering [2]. The first one, an agricultural monitoring case, illustrates a frequent remote sensing application. The second one, on multi-temporal tree clear-cuts detection, is used to show the capacity of our method to focus on only one cluster.

1) Crops dataset: An area located near Toulouse (South-west France), 1000×1000 pixels in size, was selected for this study. The dataset represents 11 multispectral images sampled over a period of eight months in 2007. One of the images in the time series is presented in Figure 3a. The multispectral images (green, red, and near-infrared bands) are captured by



(a) Image (b) Class Labels (c) Reference Data

Fig. 3. Real-world image time-series clustering data: 12 classes, and 11 time points (t_4 displayed here).

the Formosat-2 satellite and were provided by *Centre d'Études Spatiales de la Biosphère (CESBIO) Unité Mixte de Recherche CNES-CNRS-IRD-UPS*, Toulouse, France. A random subset of the image time-series is sampled (within regions for which reference data are available, see Figure 3c). The dataset is totaling 9869 pixel time series.

2) Tree clear-cuts dataset: An area of the Vosges Mountains (Alsace, France), 724×337 pixels in size, was selected for this study, see Figure 4a. The time series is composed of 11 images sampled over 3 years. The dataset contains 10 areas of clear-cut: 8 appear at t_4 and 2 at t_8. The images are composed of one band that contains the computed NDVI (normalized difference vegetation index) values. The images were computed from Copernicus Sentinel-2 data (tile T32ULU) processed at level 2A/3A by *CNES for Theia Land Data Centre*. A subset of the area was selected by randomly sampling $40,000$ pixel time series. In the following pages, tree clear-cuts may be directly referred to as tree cuts or cuts.

B. Selected methods

In this paper, we compare our method with other existing constrained methods. In particular, we used those presented in [2] excepted Spectral methods. Indeed, such methods require a step of parameters learning (i.e. σ and eigenvectors) and so they cannot be used in a fair comparison with unsupervised methods. The methods used are :

- COP-KMEANS [11] from B. Babaki implementation [2].
- MIP-KMEANS [12] from B. Babaki implementation [3]
- CPClustering [13] from the python launcher available online [4].

More details on these methods are available in [2]. For all the methods, we use constraints provided by the expert. But obtaining them is not always easy as it is time-consuming. Therefore we used some existing active constrained strategies to generate them and as a comparison to user feedback:

- MinMax [20]: this method is a neighborhood-based approach that works in two phases. In the first phase, $Explore$, it builds k disjoint neighborhoods using the farthest-first traversal method, where k is the total number of clusters. In the second phase, $Consolidate$, MinMax incrementally expands the set of instances in the neighborhoods set by selecting the most uncertain instance from the set of remaining instances according to a MinMax criterion. Afterward, queries are submitted to the oracle until an ML is found between q and a neighborhood.
- NPU (Normalized Point-based Uncertainty) [21]: this method is also a neighborhood method, but it works in an iterative scheme. It alternates between clustering the data and adding new instances in the neighborhoods set N. It picks new instances based on a similarity metric computed from a Random Forest classifier on clustering labels and select the instance that maximizes the gain of information with the minimal expected cost of queries.

We used the public implementation in the active-semi-supervised-clustering python package of NPU and MinMax for our tests[5].

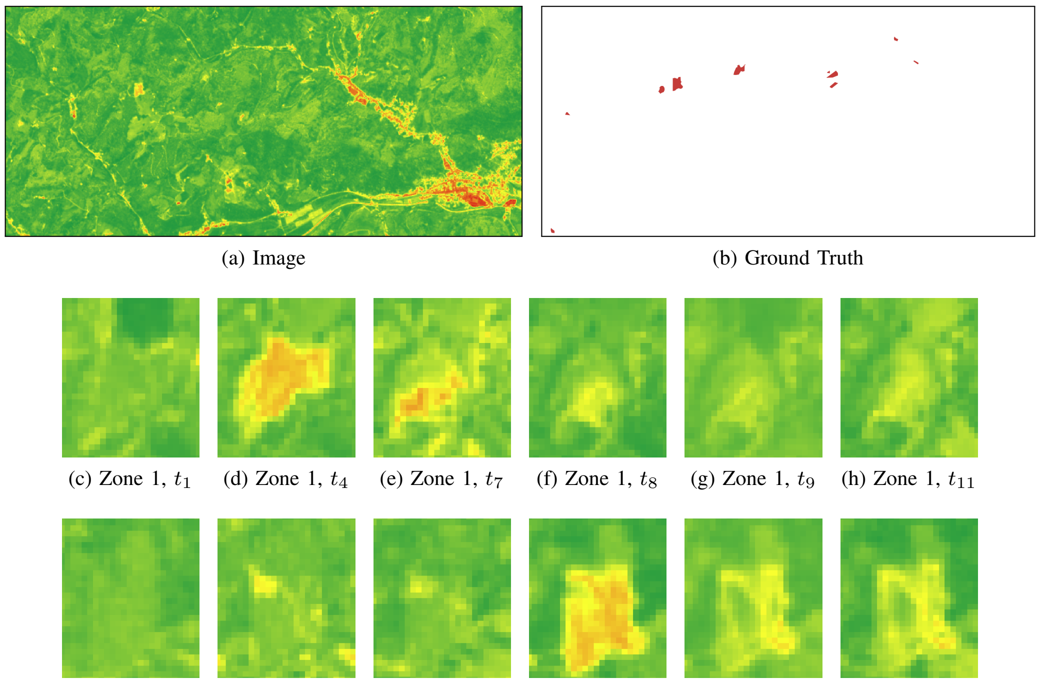
For SAMARAH and I-SAMARAH, they were configured as follow:

- 3 K-Means agents, with a number of seeds of respectively 12, 15 and 18.
- the compactness [7] is used as quality index.
- p_s, p_q and p_c have respectively a value of 0.08, 0.12, 0.8. Those are the default I-SAMARAH values for these parameters.

The choice of K-Means method for the agents was motivated by the comparison with MIP-KMEANS and COP-KMEANS methods that are based on the same clustering algorithm. It allows for better evaluation performance gains. The code of the method is available online[6] as well as a GUI for

[2]https://github.com/Behrouz-Babaki/COP-Kmeans
[3]https://github.com/Behrouz-Babaki/MIPKmeans
[4]https://icube-forge.unistra.fr/lampert/TSCC/-/tree/master/methods/Dao17

[5]https://pypi.org/project/active-semi-supervised-clustering/
[6]https://icube-forge.unistra.fr/lafabregue/JCL



(a) Image (b) Ground Truth

(c) Zone 1, t_1 (d) Zone 1, t_4 (e) Zone 1, t_7 (f) Zone 1, t_8 (g) Zone 1, t_9 (h) Zone 1, t_{11}

(i) Zone 2, t_1 (j) Zone 2, t_4 (k) Zone 2, t_7 (l) Zone 2, t_8 (m) Zone 2, t_9 (n) Zone 2, t_{11}

Fig. 4. (a) Clear-cuts NDVI time-series image (t_8 displayed here), (b) tree cut ground truth locations, (c) to (h) and (i) to (n) respective evolution of NDVI index on zone 1 and 2 sampled to the dates t_1, t_4,t_7, t_8, t_9, t_{11}.

remote sensing that allows the user to automatically launch the algorithm and add constraints[7].

Based on the results in [2] we also adapted all methods to use the DTW (Dynamic Time Warping) [22] measure with DBA (DTW Barycenter Averaging) [23] average method for the Tree clear-cuts dataset to take into account the time dimension and the Euclidean metric for the Crops dataset.

C. *User feedback and its simulation*

In our approach, we rely on expert intuitions and knowledge to give useful constraints by taking into account the previous clustering result.

For the Tree clear-cuts dataset, we were able to gather a group of remote sensing experts that agreed to participate in the experiments. During the experiment, they took the decision to stop the incremental process when they estimated that either a satisfying result is obtained, or that new constraints were not improving the result.

But in some cases, for example the Crops dataset, no experts were available as it also requires to have a good knowledge of the different crop types phenology in the selected area. So, having the required expertise is difficult because it is well known as time-consuming and fastidious. So, to get around this lack of experts we define a generic mechanism to simulate the expert. It consists in using a ground truth to generate constraints: two pixels are picked at random and a constraint is added between them of type ML if the two pixels have the same labels or of type CL otherwise. Then, we simulated the experts' feedback by only selecting constraints not satisfied by the current clustering.

[7]https://icube-forge.unistra.fr/lafabregue/Mustic

D. *Validation methodology*

Following the choices made in [2], we used the classic adjusted Rand index (ARI) to measure the similarity of the clustering results and the ground truth for the Crops dataset. The ARI score reaches its best value at 1 (perfect match), 0 for equivalent to random, and negative if it is worst.

For the Tree clear-cuts dataset, we only focus on one cluster that we want to separate from other classes. Therefore, we used the F1 score for the evaluation. The F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

However, a binary classification in such unbalanced and diverse data would not be possible in an unsupervised or a weakly semi-supervised setting. Typically, in remote sensing problems, the number of clusters in an image is unknown. Based on the expert recommendation on the expected number of evolution classes in the image series, this number is set to 15. However, the evaluation is only concerned with clusters that include clear-cuts, details on this selection will be discussed further in section V-D2. We also have to take into account the spatial context of the data, indeed in a real scenario, the expert may know a portion of the targeted data (tree cuts). However, it is not expressed in terms of the number of pixels but in the number of objects (tree cuts polygons). Hence, in our experiments, we limited all user feedback on tree cuts to only one cut area per run. For this objective, we used two different cuts to illustrate the impact of this choice on the result, zone 1 and zone 2, described in Fig 4. Note that this restriction is only used for user feedback.

Also, the F1 score is computed on one predicted class, however, for unsupervised methods, no cluster is dedicated to a specific class beforehand. Thus, we have to specify a way to select which cluster or clusters have to be selected for the evaluation. In a real case, the expert will select clusters that are close to their known example, therefore, it has to overlap a major part of this zone but also be as specific as possible. We took the set of clusters with the best ratio between pixels inside and outside the selected zone, that covers at least half of the selected zone. It allows the inclusion of small but highly representative clusters.

We also took into consideration the number of detected cuts. In a real case, the expert is satisfied if all cuts are detected, even partially. This is more useful than a result where only a portion of the cuts is detected even if their contour is well defined.

E. *Ground truth used in experiments*

For the crops, the ground truth is extracted from the RPG (Registre Parcellaire Graphique) that contains farmer's declaration of crops grown made to the European Environment Agency for the Common Agricultural Policy. This experiment is subject to a specific setup as no expert was available for the annotation process (see section IV-C).

For the Tree clear-cuts, the clear-cuts ground truth was manually created by remote sensing experts. It was also validated in the field by forestry rangers, see Figure 4b. However, the experts involved in the experiments did not have knowledge of the ground truth before or during the experiment.

V. Experiments

A. *Experiment levels*

A semi-supervised method is usually evaluated by comparing it to other state-of-the-art methods. However, in our case, we want to demonstrate that our method benefits from both allowing the user to give feedback and from its capacity to incrementally take them into account.

To show that it is the combination of these two factors that make this framework relevant, we will test the effect of each element of the framework separately. We conduct different levels of experiments on the two datasets to determine:

- Level 1: the performance of unconstrained clustering (section V-B)
- Level 2: the performance of using standard constrained clustering method (i.e. non-incremental) with randomly selected constraints (section V-B)
- Level 3: the performance with user feedback and non-incremental methods (section V-C)
- Level 4: the performance with randomly selected constraints and an incremental method (section V-D1)
- Level 5: the performance with user feedback and an incremental method (section V-D2)

For all experiments, the reported results are the average over 10 runs. Also, all iterative/incremental experiments are launched from the initial states computed in level 1. In the following pages, we will speak about I-SAMARAH initial state as the result after a complete run of unsupervised SAMARAH. An iteration i refers to the result after an initial state followed by i repetitions of the sequence user feedback followed by a complete I-SAMARAH run.

B. *Unconstrained (level 1) vs Constrained clustering (level 2)*

The first step in our experimental process is to evaluate if the methods benefit from constraint addition. To do this, we carried out the experiments from level 1 and level 2. For the Crops dataset, we used the results reported in [2].

For level 1 we run each method without using constraints to establish an unsupervised baseline. The results for each dataset are displayed in Table I. For this experiment, we also reported the constraint satisfaction rate computed with respect to 50% fraction sets (see level 2 below). It denotes the portion of constraints that are satisfied by the method even if constraints are not provided, this is referred to as the method's consistency.

TABLE I
LEVEL 1: UNCONSTRAINED AVERAGE PERFORMANCE AND SATISFACTION RATE PER DATASET. EACH DATASET'S BEST PERFORMANCE IS BOLD

	Crops		Tree clear-cuts	
Method	ARI	Sat.	F1 score	Sat.
COP-KMEANS	0.420	0.807	0.104	0.917
MIP-KMEANS	0.407	0.803	**0.107**	**0.920**
CPClustering	**0.681**	0.413	0.106	0.877
SAMARAH	0.463	**0.817**	0.036	0.916

The results show a high consistency, especially for the Tree clear-cuts dataset. This illustrates that constraints chosen at random will not bring much knowledge to the semi-supervised methods, which may lower their effect.

Then, for level 2 we add constraints in the clustering process. For constraints generation, we followed the protocol used in [2]. Constraints were generated by taking pairs of points randomly and generating a must-link or cannot-link constraint depending upon whether they belonged to the same class or not. Different sizes of constraint sets were considered: 5%, 10%, 15%, and 50% of the number of points N in the dataset (they represent a very small fraction of the total number of possible constraints, which is $\frac{1}{2}N(N-1)$). Ten repetitions of each constraint set size were generated, as such, each experiment was repeated ten times, each with a different random subset of constraints (all algorithms are evaluated using the same random constraint sets).

For the Tree clear-cuts dataset, we have to take into consideration that we only want constraints that include tree clear-cuts, as we don't know the labels for other classes. Thus, for random constraints, we first pick a pixel in a tree clear-cut and the other one is picked randomly among other tree and non-tree pixels.

The results are reported in table II, they do not include the satisfaction rate as all methods satisfy all constraints by construction, with the exception of SAMARAH that have an

TABLE II

Level 2: Average performance of constrained clustering with different fractions of random constraints. The best performance for each constraint fraction is highlighted in bold.

	Crops (ARI)				Tree clear-cuts (F1 score)			
Method	5%	10%	15%	50%	5%	10%	15%	50%
Cop-Kmeans	0.406	0.314	0.443	0.369	0.104	0.098	**0.124**	0.112
Mip-Kmeans	0.428	0.416	0.431	0.532	**0.112**	**0.109**	0.105	0.114
CPClustering	0.650	0.562	0.542	0.510	0.099	0.103	0.105	0.096
Samarah	**0.691**	**0.682**	**0.714**	**0.702**	0.102	0.097	0.108	**0.121**

average satisfaction rate of 0.86 for Crops and 0.97 for Tree clear-cuts. It can be observed beforehand that the methods do not necessarily benefit from constraints' information. This is a known effect that has already been shown and studied [14].

Mip-Kmeans and Samarah benefit from constraints addition, even though the gain for Mip-Kmeans is only significant with the 50% fraction. For Cop-Kmeans and CP-Clustering, they see their results deteriorate with constraints, especially CPClustering that have an ARI performance that decreases inversely with the number of constraints. It was highlighted in [2] that the most consistent improvements in ARI are observed with methods that do not guarantee constraint satisfaction (Samarah in our case).

For Tree clear-cuts, the results are more mitigated. Samarah is the only method to benefit largely from constraints, but it is also the method that has the lower unconstrained result. Overall, no method really gives good results that may justify the addition of constraints in the process, especially with the number of involved constraints (5% of 40000 pixels is already totaling 2000 constraints).

Moreover, the performance seems not correlated to the number of constraints, with the exception of CPClustering on Crops where it is negative.

C. User feedback and non-incremental methods (level 3)

For level 3, we propose to integrate user feedback, but also to compare it to the NPU and MinMax active constraints selection methods. However, as a first step, we use the standard constrained clustering method (i.e. without I-Samarah). For NPU and user feedback, that give constraints based on an existing clustering, the process of giving constraints was repeated for 4 iterations. This process consists in analyzing the previous clustering, an unconstrained for the first iteration, then querying constraints, and launching a constraints clustering with the new constraints. The constraints are cumulated from one iteration to the other. For user feedback, the number of constraints was limited to 6 annotations per iteration. For Tree clear-cuts, the total number of constraints varies between 13 and 21 constraints, with an average of 16 constraints. For Crops, it is fixed for a total of 24 constraints. For NPU we fixed the number of queries to 10 per iteration. For MinMax only one iteration is done as it does not depend on a previous result but with a total of 40 queries. For this experiment, the reported average is also computed over 10 runs with the exception of user feedback for Tree clear-cuts with 5 runs for each Zone.

TABLE III

Level 3: Average ARI for constrained clustering with different constraints strategies on Crops dataset. The best performance for each constraint strategy is highlighted in bold.

	Crops (ARI)		
Method	User feedback	NPU	MinMax
Cop-Kmeans	0.424	**0.427**	0.317
Mip-Kmeans	**0.533**	0.521	0.389
CPClustering	**0.649**	0.506	0.483
Samarah	**0.715**	0.587	0.431

TABLE IV

Level 3: Average F1 score of constrained clustering with different constraints strategies on Tree clear-cuts dataset. The best performance for each method is highlighted in bold. (U.f.: User feedback)

	Tree clear-cuts (F1 score)			
Method	U. f. - Zone 1	U. f. - Zone 2	NPU	MinMax
Cop-Kmeans	0.096	**0.105**	0.084	0.045
Mip-Kmeans	**0.133**	0.129	0.076	0.076
CPClustering	**0.138**	0.097	0.108	0.082
Samarah	**0.151**	0.126	0.119	0.081

The results are reported in table III for the Crops dataset and table IV for the Tree clear-cuts dataset. Two elements can be pointed out from these results:

First, similarly to random constraints, the methods do not necessarily benefit from constraints' information even when they are selected from a criterion. Samarah is again the method that benefits the most from constraints, even though the difference is not always important.

Second, user feedback is the strategy that seems to be the most consistent, especially if we do not consider Cop-Kmeans which performs poorly with all constraint strategies. But more surprisingly NPU and MinMax perform poorly, especially MinMax which gives even worst result than in the unconstrained setting. This may be explained by the way these two methods operate as they rely on an exploration mechanism that tries to find the most informative instance. To determine informativeness they use a criterion that favors the most uncertain instances that are also the ones that are far from each other in the data space. Thus, these methods tend to focus on instances from clusters that have high overall inertia. For tree cuts that represent only a small fraction of the dataset, these methods are unlikely to reach the granularity to extract it. For crops, it tends to focus on noisy parts of the dataset (e.g. road or hedge crossing the field).

In figures 5c and 5d we display four examples of CL constraints gave by the expert on the Tree clear-cuts dataset. In both negative examples (not a tree cut) we can observe a sudden decrease in NDVI. This decrease can easily be confused with tree cuts, but for negative examples, the NDVI increases rapidly afterward or drops multiple times. This evolution is typical of meadows or crops that regrow faster than forests and are often harvested multiple times in a two years period. These constraints illustrate that this approach allows the expert to easily identify good opposite examples.

D. I-SAMARAH*: incremental method (level 4 and 5)*

User feedback gave the best results, however in the level 3 experiment we launched the next iteration with the new constraint set but without taking into account the previous clustering itself. I-SAMARAH aims to answer this problem, but Beforehand, we want to measure the benefit from I-SAMARAH with random constraints.

1) Randomly selected constraints and an incremental method (level 4): For level 4, we fed 6 constraints selected randomly, similarly to level 2, to I-SAMARAH for a total of 4 iterations (1 unconstrained run followed by 4 constrained incremental runs). The results are reported in table V.

TABLE V
LEVEL 4: AVERAGE PERFORMANCE OF CONSTRAINED CLUSTERING WITH DIFFERENT FRACTIONS OF RANDOM CONSTRAINTS AT DIFFERENT PROCESS STEPS (ITR. : ITERATION).

Dataset	Initial	Last itr.	Best itr.
Crops (ARI)	0.463	0.711	0.721
Tree cuts (F1 score)	0.036	0.109	0.113

The observed results show that the performance does not really differ from SAMARAH with random constraints. The final result, after 4 iterations, is reported in the "Last itr." column, but we also reported the best result from all the iterations in the "Best itr." column. These two columns differ which implies that the performance may lower from one iteration to another. It leads to performance comparable to level 2 but with a way a smaller number of constraints. However, for Tree clear-cuts, the benefit seems still negligible.

2) User feedback and an incremental method: For this last level, we combine user feedback with I-SAMARAH. For the Crops dataset 10 runs are launched with simulated user feedback and for tree clear cuts 5 runs for each zone are launched with expert feedback and constraints are cumulated between each run. In the end, we have a total of 24 constraints for Crops. For Tree clear-cuts, this number is determined by the choice of the expert with a restriction fixed at a maximum of 6 constraints.

The overall results in Table VI show a clear increase when I-SAMARAH is used with user feedback with a significant margin. This is even the case for Crops, where we only simulated this expertise. We also reported the progression after one supplementary iteration is launched by the user for the Tree clear-cuts. One can observe that the result does not necessarily improve with a new iteration, confirming in a real case scenario the results observed on Crops and in the level 4 experiment. It even decreases in our case, despite more than a 20% increase in the number of constraints, going from an average of 14.5 before the supplementary iteration to 17.9 after. Usually, it is linked to a lower consistency between constraints, we ask the method to put together instances that are too dissimilar or to take into account outliers. During our experiment the experts tried to remove the remaining noise or include all the selected zone pixels, in consequence, the remaining pixels will be the ones that are difficult for the algorithms. In terms of the number of detected cuts, we have on average 9.8 over 10 cuts detected (only one run on zone 2 missed two cuts). The F1 score difference is explained by more false positives pixels and tree cuts that are only partially detected. Results examples are displayed in figure 5.

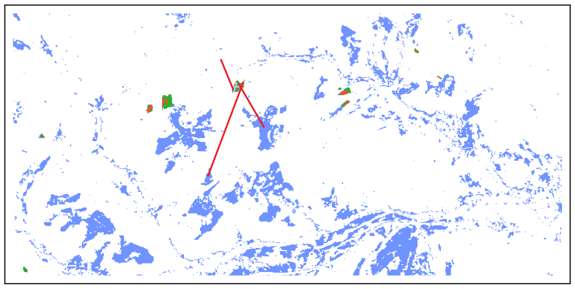
TABLE VI
LEVEL 5: EVOLUTION OF AVERAGE F1 SCORE ON 5 I-SAMARAH RUNS FOR TREE CLEAR-CUTS (ITR.: ITERATION, CST.: CONSTRAINTS).

Dataset	Initial	#itr.	#cst	Last itr.	One more itr.	Best itr.
T.c.-Zone 1	0.036	2.6	13.4	0.411	0.301	0.416
T.c.-Zone 2	0.035	3.6	15.6	0.202	0.184	0.212
Crops	0.463	∅	24	0.773	∅	0.785

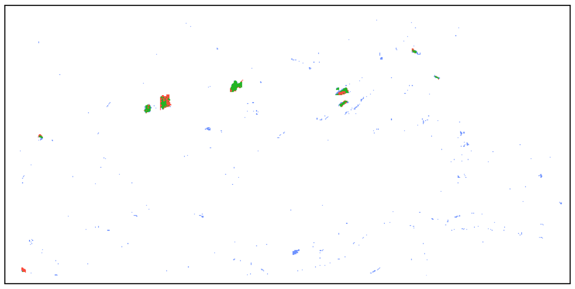
VI. CONCLUSIONS AND PERSPECTIVES

We have introduced, I-SAMARAH, a clustering method that supports user feedback in an incremental way. I-SAMARAH uses constraints to guide the clustering in an exploratory process. This approach allows I-SAMARAH to efficiently modify the existing clustering to incorporate the expert's background knowledge through constraints. Our experiments demonstrate the effectiveness of our method in terms of quality compared to state-of-the-art constrained clustering methods, but also in terms of user investment compared to supervised methods. Therefore, this method can be an alternative to supervised methods, offering a trade-off between quality and annotation time, or also help the constitution of a training set.

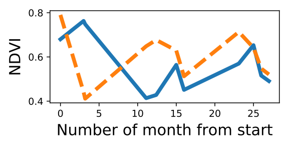
The experiments have also shown that, even if I-SAMARAH benefits more from constraints than compared constrained methods, the results deteriorate when the number of iterations/constraints is too high. This subject should undergo further investigation, but it is likely to be related to the over-constraint problem as K-means learning agents struggles to properly learn the structure of the data. Two approaches may contribute solving this problem. First, the expert may benefit from a mechanism that helps him/her to select effective constraints, either to suggest new constraints, but also when deciding if some constraints should be kept from one iteration to the other. Second, this could be mitigated by using other learning agents that take more advantage of the time and space dimensions and have the capacity to learn a more suitable data representation. It also has to be noted that this method is efficient for the expert when the clustering result can be clearly visualized. It makes this method particularly suited to the remote sensing domain but it may not be the case for other types of datasets.



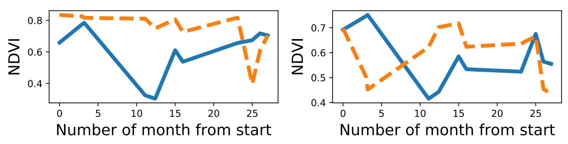
(a) Initial result with constraints set on zone 1



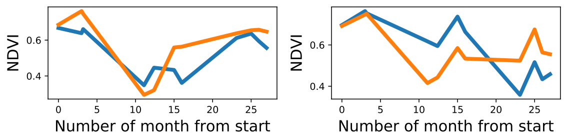
(b) Result with same constraints on zone 1



(c) CL profile examples on meadow



(d) CL profile examples (others)



(e) ML profile examples

Fig. 5. Results of I-SAMARAH with cumulative strategy, in green true positives, in red false negatives and in blue false positives. In (a) red lines represent CL constraints and cyan lines ML constraints. Only the first iteration constraints are displayed. (c) represent the final result after multiple iterations. (c) and (e) represent constraints profiles, dashed lines represent non tree cuts and each tick is spaced according to the date of capture.

References

[1] M. U. Ali, W. Sultani, and M. Ali, "Destruction from sky: Weakly supervised approach for destruction detection in satellite imagery," *ISPRS Journal of Photo. and Rem. Sen.*, vol. 162, pp. 115–124, 2020.

[2] T. Lampert, B. Lafabregue, T.-B.-H. Dao, N. Serrette, C. Vrain, P. Gançarski, *et al.*, "Constrained distance-based clustering for satellite image time-series," *JSTAR*, vol. 12, no. 11, pp. 4606–4621, 2019.

[3] B. Lafabregue, J. Weber, P. Gançarski, and G. Forestier, "Deep constrained clustering applied to satellite image time series," *Proceedings of the 2019 MACLEAN workshop*, 2019.

[4] Z. Li, J. Liu, and X. Tang, "Constrained clustering via spectral regularization," in *2009 CVPR*, pp. 421–428, IEEE, 2009.

[5] B. Settles, "Active learning literature survey," tech. rep., University of Wisconsin-Madison Department of Computer Sciences, 2009.

[6] T. Van Craenendonck, S. Dumančić, E. Van Wolputte, and H. Blockeel, "Cobras: fast, iterative, active clustering with pairwise constraints," *arXiv preprint arXiv:1803.11060*, 2018.

[7] G. Forestier, P. Gançarski, and C. Wemmert, "Collaborative clustering with background knowledge," *Data & Knowledge Engineering*, vol. 69, no. 2, pp. 211–228, 2010.

[8] P. Jonsson and L. Eklundh, "Seasonality extraction by function fitting to time-series of satellite sensor data," *IEEE transactions on Geoscience and Remote Sensing*, vol. 40, no. 8, pp. 1824–1832, 2002.

[9] G. Roerink, M. Menenti, and W. Verhoef, "Reconstructing cloudfree ndvi composites using fourier analysis of time series," *International Journal of Remote Sensing*, vol. 21, no. 9, pp. 1911–1917, 2000.

[10] S. Basu, I. Davidson, and K. Wagstaff, *Constrained clustering: Advances in algorithms, theory, and applications*. CRC Press, 2008.

[11] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, *et al.*, "Constrained k-means clustering with background knowledge," in *ICM*, vol. 1, pp. 577–584, 2001.

[12] B. Babaki, "MIPKmeans," 2017.

[13] T.-B.-H. Dao, K.-C. Duong, C. Vrain, *et al.*, "Constrained clustering by constraint programming," *Artificial Intelligence*, vol. 244, pp. 70–94, 2017.

[14] I. Davidson, K. L. Wagstaff, and S. Basu, "Measuring constraint-set utility for partitional clustering algorithms," in *ECML-PKDD*, pp. 115–126, Springer, 2006.

[15] B. Settles, M. Craven, and S. Ray, "Multiple-instance active learning," in *Advances in neural information processing systems*, pp. 1289–1296, 2008.

[16] D. Cohn, R. Caruana, and A. McCallum, "Semi-supervised clustering with user feedback," *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, vol. 4, no. 1, pp. 17–32, 2003.

[17] I. Davidson, S. Ravi, and M. Ester, "Efficient incremental constrained clustering," in *SIGKDD*, pp. 240–249, 2007.

[18] G. Forestier, C. Wemmert, and P. Gancarski, "Towards conflict resolution in collaborative clustering," in *2010 5th IEEE International Conference Intelligent Systems*, pp. 361–366, IEEE, 2010.

[19] P. Gançarski, B. Lafabregue, A.-D. Salaou, and H. Vernier, "Fodomust-une plateforme de clustering collaboratif sous contraintes incrémental de séries temporelles.," in *EGC*, pp. 507–514, 2020.

[20] P. K. Mallapragada, R. Jin, and A. K. Jain, "Active query selection for semi-supervised clustering," in *2008 19th ICPR*, pp. 1–4, IEEE, 2008.

[21] S. Xiong, J. Azimi, and X. Z. Fern, "Active learning of constraints for semi-supervised clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 43–54, 2013.

[22] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE transactions on acoustics, speech, and signal processing*, vol. 26, no. 1, pp. 43–49, 1978.

[23] F. Petitjean, A. Ketterlin, and P. Gançarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition*, vol. 44, no. 3, pp. 678–693, 2011.