KEO :
Knowledge-centred Earth Observation

Work Package:
Support for Detailes Design for Time-Series

KEO
Issue: 1.0
Date: /2005

Document Signature Table

	Name	Function	Signature	Date
Author	P.HEAS			
Verification				
Quality Assurance				
Approval				

Document Change Record

Issue/Revision	Date	Reason for Change	Changed Pages/Sections
			-

Distribution List

Internal Distribution	
Name	**No. Copies**

External Distribution		
Company	**Name**	**No. Copies**

TABLE OF CONTENTS

1 INTRODUCTION

This work is aimed at providing a consultancy on the complex problem of mining image time-series of satellite images. This can be done in several ways, but we will concentrate on the most promising one which consists in modelling hierarchically the data information content in a Bayesian framework. Therfore a concept for spatio-temporal information mining is introduced in section 2. The relevance of the approach is demonstrated in section 3 with several examples of probabilistic retrieval of spatio-temporal patterns. The implementation of this concept results in a system composed of an automatic processing chain and a interactive learning module. Details on the technical aspects of the system are provided in section 4 and in the appendices.

2 CONCEPT FOR SPATIO-TEMPORAL INFORMATION MINING

Authors: Patrick Héas / Mihai Datcu

2.1 PROBLEM CHARACTERIZATION

2.1.1 Satellite image time-series

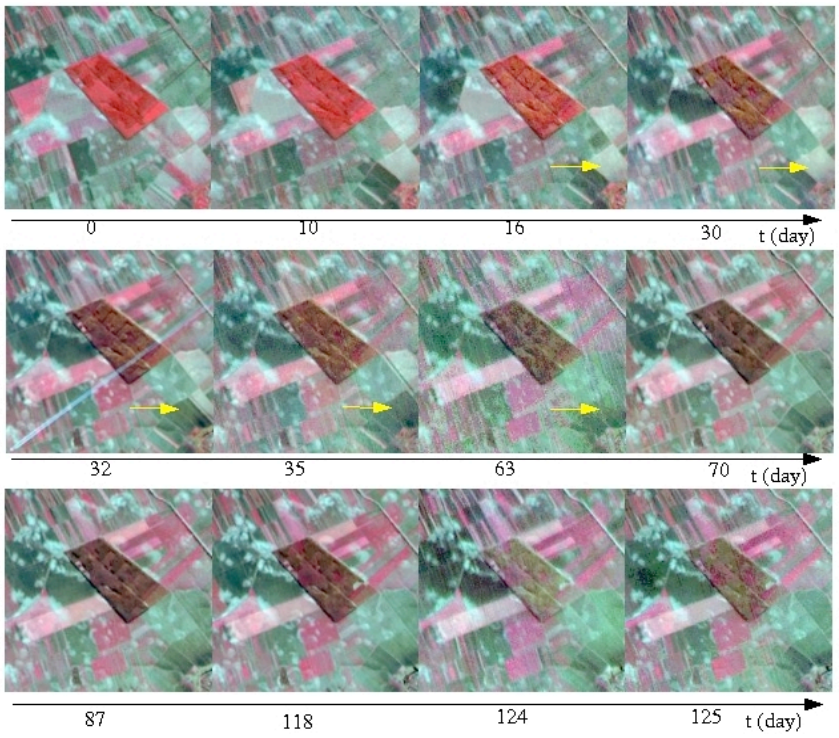


Figure 1*: SITS contain many spatio-temporal structures. The yellow arrow points out a ploughing phenomenon occurring in the ADAM SITS. We can also see a plane occlusion in the image acquired 32 days after the first acquisition (14 November 2001), as well as the smooth evolution of the forest cover.*

Nowadays, huge quantities of satellite images are available from many different Earth observation sites. Moreover, thanks to a growing number of satellite sensors, the acquisition frequency of a same scene is permanently increasing. Furthermore, the high spatial resolution of the sensors gives access to detailed image structures. Thus, opportunities to compose high resolution SITS are growing and the observation of precise spatio-temporal structures in dynamic scenes is getting more and more accessible.

A SITS composed of SPOT multispectral images containing 2000x3000 pixels is partially visible in figure 1. The spatial resolution is 20 meters. The acquired scene is a rural area located in the East of Bucharest (Romania). The acquisition campaign was driven in order to provide remote sensing data for the *Data Assimilation for Agro-Modeling (ADAM)* project. The SITS was obtained by daily acquisition and by filtering out images presenting a cloud or a snow cover above the project test sites. This selection procedure resulted in 38 images irregularly sampled in time, which were acquired over a period of 286 days. Figure 2 displays the irregular sampling of the SITS. The images were then made superposable and a radiative transfer model was applied to produce reflectance measurements. The ADAM project SITS is available on-line [1].

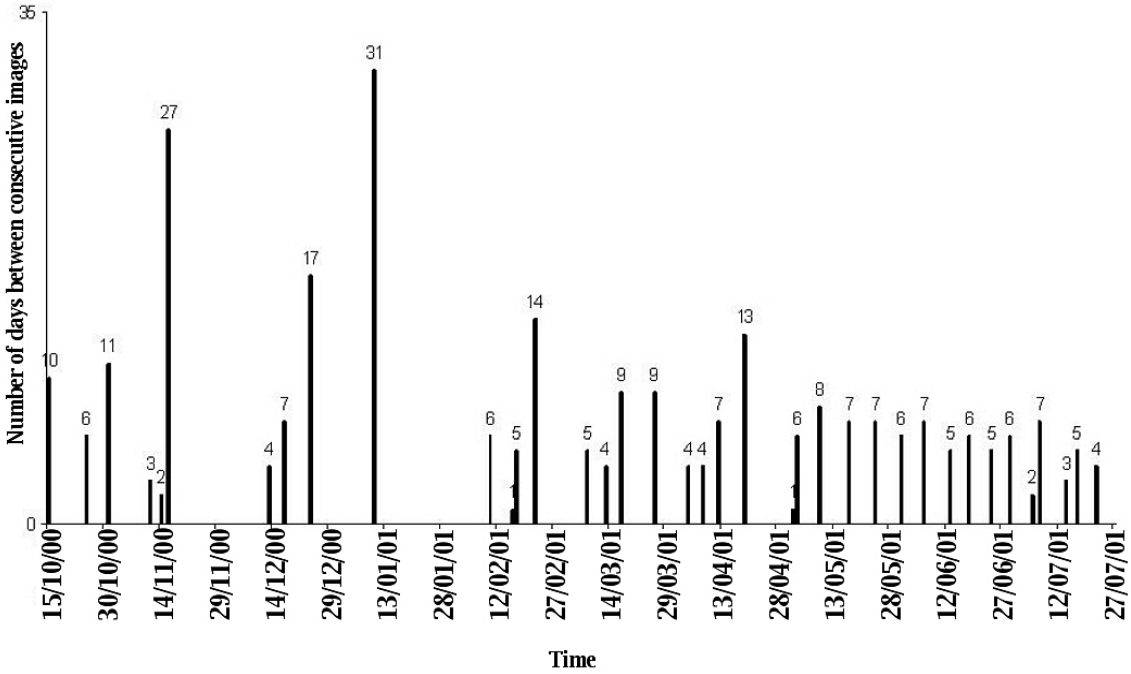


***Figure 2** : Acquisition dates of cloud free images in the ADAM database. The horizontal axis represents time which is irregularly sampled while the vertical axis represents the time difference between consecutive acquisition dates..*

SITS are complex objects possessing a rich information content. They contain numerous and various spatiotemporal structures. For example in rural scenes, one can observe the growth and the maturation of cultures, their harvests, evolutions of ploughland, river floods, etc. Near urban areas, car and plane occlusions are frequent but there are also evolving constructions, pollution phenomenon, etc. Some examples of spatio-temporal structures are pointed out in figure 1.

The analysis of spatio-temporal structures are useful to understand complex evolutions which concern various domains such as agriculture, forest monitoring, ecology, hydrology, urbanization, etc. But our capacity to store these large volumes of data has exceeded our ability to access the broad variety of information contained in it. Indeed, limited tools exist to exploit this huge potential of information.

Change detection, monitoring and validation of physical models by data assimilation constitute the most used analysis for information extraction in SITS. The methods developed in these fields are complicated and dedicated to specific applications. Although these techniques are efficient, together they represent a limited range of applications. Nevertheless, one may be interested in finding a specific forest cover evolution or in detecting wheat harvests occurring during a given period. Until now, only few methodshave been developed. They mainly focus on low resolution images regularly sampled in time such as meteorological data [2][3][4]. Thus, in order to adapt to a broader range of application and to have access to the variety of information contained in SITS, collaborative and generic methods are needed.

2.1.2 Information Mining

Large volumes of data are important resources. But to be relevant, users must be able to interpret the data information content. Understanding this huge quantity of data, which may be complex and multidimensional, can represent laborious work for users. Images are particularly complex objects possessing rich information contents. A manual analysis of associations and relations among images is not feasible. Furthermore, the usefulness of such an analysis may be restricted to a particular application. But there is a broad diversity of application domains and it is not possible to produce a specific analysis for each one of them. Generic analysis methods are needed to respond specifically to the needs of each application domain.

In summary, in many fields, there is a real need to transform growing databases into knowledge. The objective of information mining is to solve this problem, by adapting the data information content to the users' needs. Information mining can be defined as the non-trivial process of analyzing data in the perspective of discovering implicit but potentially useful information. The discovered information can be for instance patterns, association rules, causal effects, changes, anomalies, etc. An information mining perspective enables content based retrieval, knowledge discovery and data understanding.

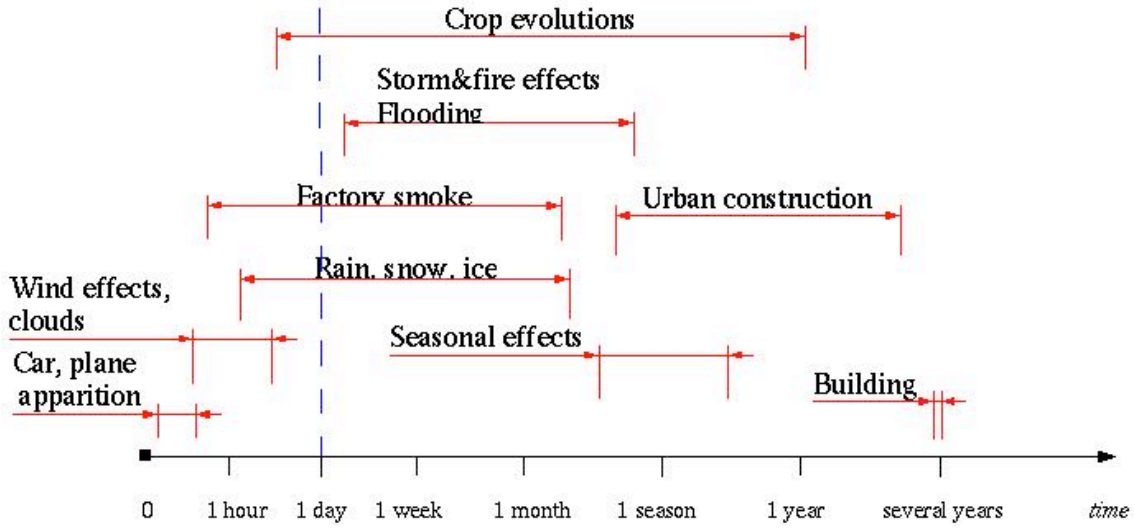


Figure 3 *: Phenomenon process changes which occur in a dynamic scene have different time-scales. For example, plane occlusions are evanescent objects over short time-periods whereas buildings are stationary objects over long time-periods. Phenomena process which are changes relevant to crop evolutions, such as the growth of crops or their harvests, possess various time-scales. Spatio-temporal objects with time-periods below the blue dashed line, might possess higher frequencies in their spectrum than the sampling limit frequency.*

The information mining problematic can be understood as a communication channel problem with : on one hand the database representing the information source and, on the other a user representing the receptor. Along the channel, the data is hierarchically processed, inducing a signal representation followed by a semantic representation. The signal representation is obtained by extracting information from the data by stochastic modeling of the signal. In such a case, extracted information is described with a particular model vocabulary, which is unmeaningful for users. The semantic representation is obtained by modeling the users' semantics. In such a case, information is described with a vocabulary and a syntax natural to users.

The goal of information mining is to bridge the semantic gap, that is to say, to minimize the loss of information between information available through the semantic representation, and information obtained by a direct data inspection.

Before going into the kernel of this paper which is the description of a SITS information mining concept, let us motivate our approach by pointing out some difficulties for SITS modeling.

The analysis of spatio-temporal structures in SITS is particular. Indeed, heterogeneous temporal and spatial scales characterize structures. Figure 3 illustrates the variety of temporal scales attached to structures. Spatial scales of structures are also very different. Thus, SITS modeling methods should capture information at various scales. However, a pixel-localized time-series analysis is generally not appropriate to characterize high resolution SITS structures. For the ADAM database in particular, the superposability difficulties, the irregular sampling, and the sampling limit frequency, prevent a pixel-localized stochastic modeling. The dashed vertical line in figure 3 illustrates this limit. It discriminates objects which possess higher frequencies in their spectrum than the sampling limit frequency. To fight against these constraints, analysis at an object level may be more robust against noise and superposability errors. Moreover, it may enable an investigation below the frequency limit by using contextual information. For instance, the behavior of the smoke of a factory, which is an object evolving in space according to the wind effects and in time according to the factory activities, can be modeled in a more efficient way using its spatial context rather than a space localized time-series analysis.

To respond to the problematic of information mining in SITS, we present in the following section a Bayesian hierarchical modeling of SITS information content. The different inference steps of the hierarchy are hereby detailed : we present in section 3 an unsupervised learning procedure which results in a graph of dynamic cluster trajectories, and in section 4 we detail an interactive learning procedure which results in a semantic labeling of spatio-temporal structures. The graph of trajectories and the semantic labels constitute the signal and the semantic representations of the SITS information content.

2.2 BAYESIAN HIERARCHICAL MODELING OF SITS INFORMATION CONTENT

In order to build an information mining system for dynamic scene understanding which is free from the application specificity and which enables its open use in almost any scenario, we use a Bayesian hierarchical model made up of 6 different levels (Fig. 4). The model links the information source D, which represents a SITS comprising spatio-temporal structures, to the different users' semantics $\mathcal{A}_\nu$.

The hierarchy is defined by the relation between the random variables

$$D \rightarrow \Theta \rightarrow \Psi \rightarrow C \rightarrow \mathcal{G}_k \rightarrow \mathcal{A}_\nu \qquad (1)$$

where D, θ, ψ, C, $\mathcal{G}_k$ and $\mathcal{A}_\nu$ are refined levels of information representation. As it is a Bayesian hierarchical model, the inference of a higher level in the hierarchy depends on the adjacent lower level

and, conditionally to the latter, is independent of all other lower levels. Thus, given some data, we infer the most likely model by maximizing the joint distribution

$$p(\boldsymbol{D} \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \boldsymbol{\psi})p(\boldsymbol{\psi} \mid \boldsymbol{C})p(\boldsymbol{C} \mid \boldsymbol{\mathcal{G}_k})p(\boldsymbol{\mathcal{G}_k} \mid \boldsymbol{\mathcal{A}_\nu})p(\boldsymbol{\mathcal{A}_\nu}), \qquad (2)$$

Hence, the levels of the hierarchy are learned independently by applying Bayesian inferences or inferences based on entropic measurements on families of stochastic models. Moreover, for each learning step, we can incorporate prior knowledge by using Bayes rule.

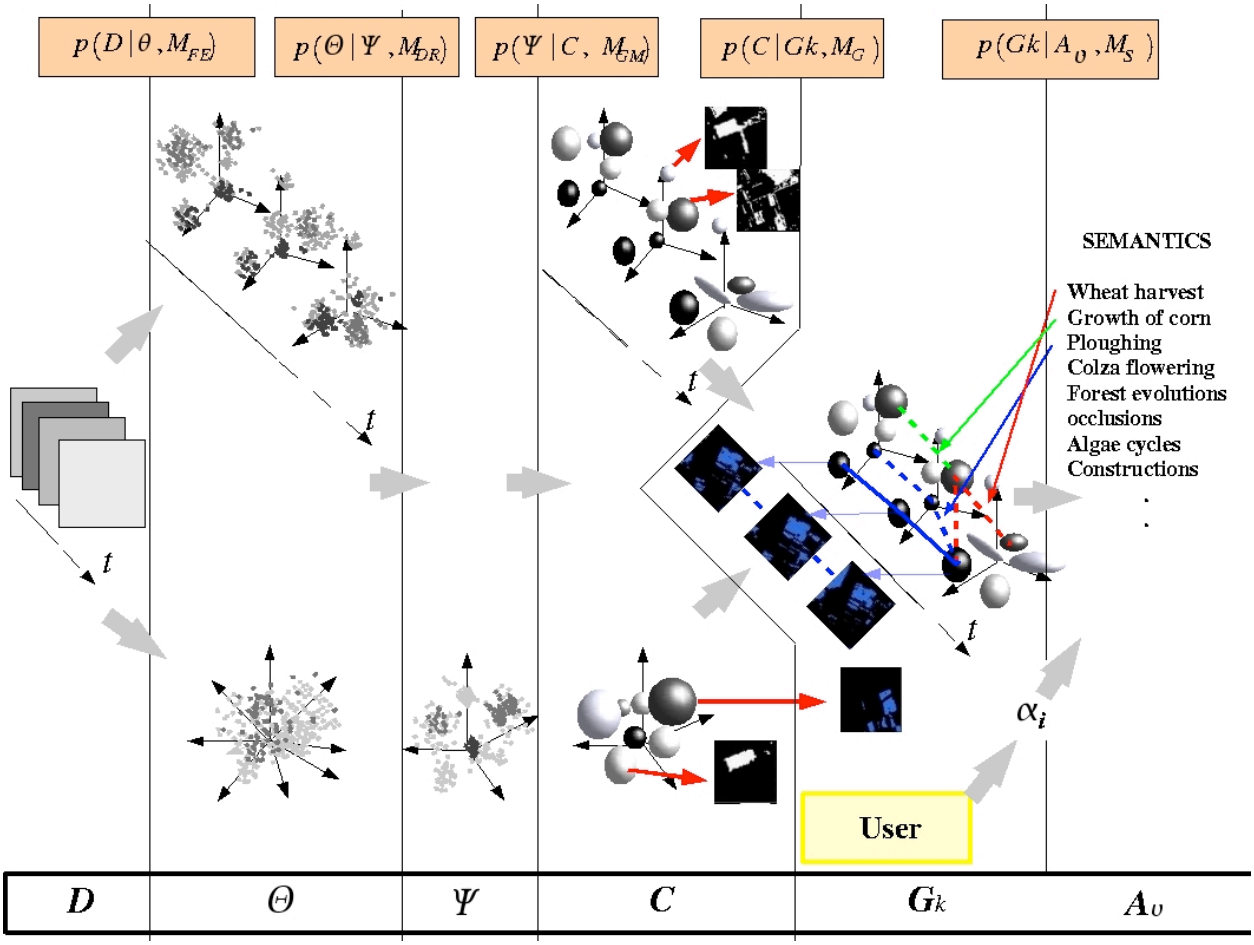


Figure 4 : *Hierarchical modeling of SITS information content. The hierarchy enables users to link spatio-temporal structures to their specific interests. First, primitive SITS features* $\boldsymbol{\theta}$ *are extracted from the data* $\boldsymbol{D}$*. Two complementary representation are induced by using the TL feature spaces and the MT feature space. Next, dimension reduction techniques are applied and result in projected features* $\boldsymbol{\psi}$*. Then, the feature distributions are learned and clusters* $\boldsymbol{C}$ *and classes are discriminated. The unsupervised learning procedure is finally achieved by inferring graphs* $\boldsymbol{\mathcal{G}_k}$ *and dynamic classifications, which code the data structures. By interactive learning, the user interests* $\boldsymbol{\mathcal{A}_\nu}$ *are linked to the graphs and semantic labels are assigned to spatio-temporal structures.*

We decompose the hierarchy into two parts :

- unsupervised learning $D \rightarrow \mathcal{G}_k$: for a particular SITS realization $\boldsymbol{D}$, we infer a collection of graphs $\boldsymbol{\mathcal{G}_k}$ representing dynamic cluster trajectories coding spatio-temporal structures;
- interactive learning $\mathcal{G}_k \rightarrow \mathcal{A}_\nu$: the collection of graphs $\boldsymbol{\mathcal{G}_k}$ are linked to a collection of users' semantics $\boldsymbol{\mathcal{A}_\nu}$; thus we induce a semantic labeling of spatio-temporal structures of SITS.

The graph inference, which requires a significant computational cost, is an application-free learning procedure. Based on this objective representation, the semantic labeling, which is interactive, learns user-specific interests using positive and negative examples.

Before detailing the levels of the hierarchy, let us introduce several spaces for the SITS representation. Image time-series are stochastic processes which are usually represented in a multidimensional space comprising two spatial axes r, a time axis t, and several feature components θ (radiometric values, texture parameters,...). Since the features are assumed independent, the analysis of the multidimensional space is done independently on the different types of features. In such a multidimensional space, the signal denoted by is represented by a cloud of points. This is the natural space for the SITS representation. But, SITS possess several other representations which lead to various interpretations. We hereby introduce as follows, several spaces for the SITS representation which are used in the proposed information mining concept.

The space formed by the feature components θ and the time axis t is called the ***dynamic feature space***. In this space, SITS is represented by a histogram of features evolving in time.

Considering d time samples, we denote by $\{\theta_{t_i}\}$ the d feature components localized at different times $\{t_i;\ i \in [1,...,d]\}$. We can form d different ***Time Localized (TL) feature spaces*** with the d different time localized feature components θ_{t_i}. In them, we represent SITS by a succession of histograms of features. These spaces constitute the different states of the dynamic feature space. If we group features by similarity, we obtain d different collections of K_{t_i} clusters $\{C_{t_i}^j; j = 1,...,K_{t_i}\}$. In the image space, we represent the d spatial classifications $\{C_{t_i}^j(r); j = 1,...,K_{t_i}\}$ associated to the collections of TL clusters.

The ***MultiTemporal (MT) feature space*** is a multidimensional space composed of the d TL feature components $\{\theta_{t_i}; i = 1,...,d\}$. In this space we represent SITS by a multidimensional histogram of features. If we group features by similarity, we obtain a collection of K_{MT} clusters $\{C_{MT}^j; j = 1,...,K_{MT}\}$. In the image space, we represent the spatial classification $\{C_{MT}^j(r); j = 1,...,K_{MT}\}$ associated to the collection of MT clusters. By projecting the MT clusters in the different TL feature spaces, we can decompose the MT representation and reconstruct the different states of the dynamic feature space. We denote by $C_{MT_{t_i}}^j$, an MT cluster projected in the TL feature space at time t_i.

Equipped with these SITS representations, we present the different levels of the hierarchy.

- The lowest level represents the data $\boldsymbol{D}$, which is constituted by spatio-temporal structures defined in time windows and spatial masks.

- First, by using different signal models, features $\boldsymbol{\theta}$ are extracted from the data at a pixel level for the different time locations $\{t_i\}; i = 1, ..., d$. We then induce two complementary representations by placing them in the TL feature spaces and in the MT feature space.
- Next, to fight against the ``curse of dimensionality'', we employ dimension reduction techniques to extract, from the MT feature space, interesting projections containing linear and non-linear structures. The projected features $\boldsymbol{\psi}$ are represented in a space composed of the extracted components.
- Then, the distribution of the MT projected features and the TL features are learned using a Gaussian mixture model of unknown complexity. The modeling procedure discriminates MT and TL clusters $\boldsymbol{C}$ possessing Gaussian shapes. MT and TL classifications are then produced, by mapping these clusters in the image space.
- The unsupervised learning procedure is finally achieved by inferring graphs $\boldsymbol{\mathcal{G}_k}$ coding the data structure of SITS. They model the dynamic feature space by formalizing trajectories of MT clusters through TL clusters. Additionally, a spatial constraint is introduced into the inference by using the MT and TL classes.
- By interactive learning, the users' interests $\boldsymbol{\mathcal{A}_\nu}$ are linked to the graphs which represent spatio-temporal structures. To complete this semantic labeling, parameters of a graph similarity model are interactively estimated by updating probabilities of a Bayesian network. This update is performed using a Dirichlet model with positive and negative examples provided by a user.

Further details on this hierarchical model can be found in [18].

2.3 UNSUPERVISED INFERENCE OF A GRAPH OF DYNAMIC CLUSTER TRAJECTORIES

2.3.1 Feature extraction

Dynamic scene understanding relies on the ability and robustness of information extraction from the observed data. We apply appropriate stochastic models M_{FE} to capture spatial, spectral, or geometric structures in each image of the time-series at a pixel level. These models are given as parametric data models $p(\boldsymbol{D} \mid \boldsymbol{\theta}, M_{FE})$ and assign a likelihood to a given realization of the data $\boldsymbol{D}$ for a particular value of the parameter vector $\boldsymbol{\theta}$. Examples of these image models are Gibbs-Markov random field models for textural features or the intensities of the multi-spectral images for spectral features [5]. Of course, for the latter, no sophisticated modeling is involved. The extracted features are represented in the MT feature spaces. This signal carries the global information on the time-series. But, these extracted features can also form, in the TL feature spaces, a collection of signals containing precise spatial information.

In the next two sections, we detail the analysis of the extracted features represented first in the MT feature space and then in the TL feature spaces. Finally, we use these two types of representation to model the dynamic feature space.

2.3.2 Modeling a multitemporal feature space

2.3.2.1 Dimension reduction

The MT feature space represents a space of high dimensionality since it results from the union of all the TL feature spaces. A direct application of a nonparametric procedure is severely restricted in this case, by the limitation called the ``curse of dimensionality" [6]. However, the information contained in a feature space of high dimensionality can often be represented with fewer dimensions. In remotely sensed images in particular, the spectral bands usually present redundancies. Furthermore, the phenomenon is likely to be accentuated when considering an MT feature space. Dimension reduction techniques, exploit this property to reduce the space dimension by extracting interesting projections. Assuming a model M_{DR} for the m dimensional distribution associated with n feature realizations $\boldsymbol{\theta} = \{\theta^1, ..., \theta^n\}$, the quality of the dimensional projected features $\boldsymbol{\psi} = \{\psi^1, ..., \psi^n\}$ can be evaluated by the likelihood $p(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \mathcal{M}_{DR})$. Gaussian distributions for PCA or just non-Gaussian distributions for PP are examples of these models.

Principal Component Analysis is a linear projection of a m dimensional space into a space in which the axes of the projections called principal component axes are decorrelated. Moreover, the principal components are ordered according to a variance criterion. In other words, the j^{th} eigenvector of the data covariance matrix corresponding to its j^{th} biggest eigenvalue defines the j^{th} principal component axis. This eigenvalue decomposition is more convenient than the maximization of the projection likelihood. The analysis relies on the assumption that the data has a normal distribution in the feature space.

In order to perform a dimension reduction, only l principal components with $l < m$ should be selected. To evaluate the loss of information, we use the signal energy $\mathcal{W} = \sum_{i=1}^{m} \lambda_i$ where the λ_i represent the eigenvalues of the features $\boldsymbol{\theta}$ autocorrelation matrix. Thus, selecting the l first principal components corresponds to a percentage α of restitution of the signal energy equal to $\alpha = \frac{\sum_{i=1}^{l} \lambda_i}{\mathcal{W}} * 100$.

Projection Pursuit groups dimension reduction techniques that extract linear projections which contain non-linear structures from a multidimensional space. Furthermore, under certain assumptions, the extracted components are independent from the others. The extracted components are ordered according to a criterion of non-Gaussianity evaluated by a projection index. This approach is equivalent to the maximization of the projection likelihood [7]. The analysis relies on the very general assumption that the data possesses a non-Gaussian distribution, which is in most cases, a valid hypothesis. After selecting a sufficiently large percentage of the signal energy with PCA, we apply PP on the reduced signal in order to exploit the extra compression associated with non-linear relationships.

The higher the dimensionality of the extracted projections, the richer their information contents. For example, structures in a plan can not always be detected in one-dimensional projections. We perform our analysis using a bi-dimensional projection index based on the chi-square distance. The relevance of this distance for the approximation of Kullback-Leibler divergence has been shown in [8]. In order to reveal all the maxima of the projection index, we employ an efficient stochastic optimization procedure proposed by Posse [9] combined with the structure removal technique proposed by Friedman [10].

To perform a dimension reduction, the l first independent components with $l \leq m$ are selected. P-values are employed in order to decide whether the components are the effect of noise or are really independent. Indeed, P-values are useful to determine limits which correspond to probabilities of projection independence [8]. Thus, for a given index limit called quantile, l independent dimensions are extracted and dimension reduction is achieved.

Dimension reduction techniques such as PCA and PP are able to condense the information contained in the MT feature space into a sub-space of lower dimensionality. They provide an efficient solution for parameter and model inference in multidimensional spaces with limited sample sizes. For further details on these techniques, we refer the reader to section 6.1.2 of [19].

2.3.2.2 Mixture modeling

A. Gaussian mixture models

Stochastic models are appropriate tools to learn about this multidimensional signal. A Gaussian mixture model is able to approximate efficiently, any distribution for which no prior knowledge is available and in particular multimodal distributions. Thus, a Gaussian mixture assumption is well suited to model the distribution of the l dimensional realizations $\psi = \{\psi^1, ..., \psi^n\}$ of the random variable Ψ, which are assumed independent and identically distributed. A mixture modeling procedure can infer similarities that can then be used for clustering the feature space. Components of the mixture are constituted by the grouping of similar feature points and thus, will define clusters C.

A l dimensional Gaussian mixture model M_{GM} is composed of K components weighted by $\{\pi_k\}$ with mean vectors noted as $\{M_k\}$ and covariance matrices noted as $\{A_k\}$. The quality of the model for given realizations ψ si evaluated by the likelihood $p(\psi^i \mid C, M_{GM})$. In order to perform the modeling procedure without any constraints either on the number of Gaussians present in the mixture or on their parameters, a criterion is needed to select the best model among all the possible Gaussian mixture configurations. In order to infer among a collection of models, the minimum description length (MDL) principle is applied.

B.MDL principle for Gaussian mixture modeling

For the n realizations $\psi = \{\psi^1, ..., \psi^n\}$, we choose out of a finite set of possible models $\{\mathcal{M}_1, ..., \mathcal{M}_M\}$, a model hypothesis $\mathcal{M}_j$ of distribution $p(\psi \mid \mathcal{M}_j)$ for ψ. We consider also the code length function (measured in bits) $L_{\mathcal{M}_j}(\psi)$ needed for the description of ψ under the model hypothesis $\mathcal{M}_i$. A bijection appears between the probability distribution $p(\psi \mid \mathcal{M}_j)$, and the code length function $L_{\mathcal{M}_j}(\psi)$.

Minimizing this code length, called by Rissanen, ``stochastic information complexity'', over $\mathcal{M}_j$ selects the model maximizing the Bayesian evidence. But the computational cost of this quantity is often prohibitive. A first order approximation is achieved by the so called ``2-part MDL code'' [11]. For parametric model families, this code length function noted as $L_{2P}(\psi)$ is composed of two terms : the code length necessary to encode the model and its estimated parameter $\widehat{\phi}$, and the code length

necessary to encode the data keeping in mind the model and its estimated parameters. The first description length part is induced by the model and parameter encoding using non-informative prior distributions. The second description length part is related to the model maximum likelihood $p(\boldsymbol{\psi} \mid \hat{\phi}, \mathcal{M}_j)$.

The MDL principle states that the best model among a collection of tentatively suggested ones, is the one that encodes the data with the smallest code length. To estimate the code length, we can use the stochastic information complexity or its first order approximation, the 2-part MDL code. The computation of this approximation requires a less intensive calculation procedure and is particularly convenient for mixture modeling.

On the basis of the 2-part MDL code, we derived the description length of the data for the family of Gaussian mixture models. A simplified model, neglecting the influence of surrounding Gaussian components and assuming constant variances for the Gaussians, was previously developed in [12]. A 2-part description length, derived from modeling a mixture of uncorrelated Gaussians, has been proposed by Wallace and Dowe [13]. In this paper, we extend this algorithm to the correlated Gaussian mixture model. This algorithm is to some extend, equivalent to the Bayesian Autoclass algorithm [14]. We assume the hypothesis of non-interfering Gaussians. The 2-part MDL code length for encoding the data $\boldsymbol{\psi} = \{\psi^1, ..., \psi^n\}$ using a Gaussian mixture model of K Gaussians of dimensionality l, is thus defined. Section 6.1.3.1 of [19] details how this 2-part description length is derived.

C.Optimization

The goal is to estimate K, $\{(M_k, A_k)\}$ and $\{\pi_k\}$, by minimizing $L_{2P}(\boldsymbol{\psi})$. Enumerating all configurations and evaluating the 2-part MDL code is not feasible. Instead, an optimization algorithm which evaluates the changes of the code length between two configurations rather than the code length itself is used.

The change of the code length $\Delta_{G_{k'}}(L_{2P}(\boldsymbol{\psi}))$ induced by the removal of a given Gaussian $G_{k'}$ can be expressed as a function of $G_{k'}$. We refer the reader to section 6.1.3 of [19] for the detail of this function. With this establishment, we introduce the optimization algorithm. It is composed of the following steps :

- *1 Initialization* : A initial Gaussian mixture is produced. It is composed of a high number $K^{(0)}$ of Gaussian, with parameters noted as $\{(M_k^{(0)}, A_k^{(0)})\}$ and $\{\pi_k^{(0)}\}$. The initialization is done by randomly spreading the clusters according to a Gaussian distribution of mean and variance learned from each data feature component.

- *Adaptation* : At iteration (q), we consider $K^{(q)}$ Gaussians in the mixture. An Expectation Maximization (EM) algorithm [15] is used to perform a ML estimation of the Gaussian mixture parameters $\{(\widehat{M}_k^{(q)}, \widehat{A}_k^{(q)})\}$ and $\{\widehat{\pi}_k^{(q)}\}$.

- *Selection* : For the same iteration (q), we remove the Gaussian $G_{k'}$ which induces the biggest decrease in the description length

$$\Delta_{G_{k'}}^{(q)}(L_{2P}(\boldsymbol{\psi}^n))$$

Then we increment (q) and go back to step 2. If no decrease is observed, that is to say if

$$\forall G_{k'},\ \Delta^{(q)}_{G_{k'}}(L_{2P}(\psi^n)) < 0$$

,

we do not remove any Gaussians and go to step 4.

- *Convergence* : if at iteration (q_end), no other decrease in the description length is observed, then the algorithm stops iterating step 2 and 3. We then obtain the estimated number of Gaussians $K^{(q\_end)}$, with the ML estimates of the parameters of the mixture model $\{\widehat{M}_k^{(q\_end)}, \widehat{A}_k^{(q\_end)}, \widehat{\pi}_k^{(q\_end)}\}$.

Section 6.1.3.2 of [19] details the optimization procédure.

The MT feature space is modeled according to a Gaussian mixture distribution. Thus, we learn the parameters K_{MT}, $\{(M_k, A_k)\}$ and $\{\pi_k\}$ related to the Gaussian mixture model. The modeling procedure infers similarities which are then used to cluster this multidimensional feature space. Therefore, each Gaussian G_k comprises feature points and defines a cluster C^j_{MT}. In parallel, MT spatial classes $C^j_{MT}(r)$ of the image time-series are generated.

2.3.3 Time-localized representations

2.3.3.1 Projecting the multitemporal feature space

From an initial perspective, the signal representation in the MT feature space $x_r(\theta_{t_1}, ..., \theta_{t_d})$ can be projected to enable TL representations. Consequently, each MT cluster C^j_{MT} with $j \in [1, K_{MT}]$ can be projected into d different TL feature spaces. We obtain projected MT clusters denoted by $\{C^j_{MT_{t_i}}; t_i \in [1, d]\}$. The projected clusters are representative of global behaviors decomposed in time. Furthermore, they are specific of the MT feature space modeling. Consequently, they contain information about the time evolution of the feature distribution.

2.3.3.2 Modeling the time-localized feature space

New modeling procedures can be performed directly for each of the TL representations $x_r(\theta_{t_i})$, independently from the MT feature space modeling. This procedure produces d sets of TL clusters $\{C^j_{t_i}; t_i \in [1, d]\}$ with $j \in [1, K_{t_i}]$, where K_{t_i} is the number of estimated clusters at time t_i. In parallel, spatial classes $C^j_{t_i}(r)$ are obtained. In this case, the TL clusters are defined for given time locations which are specific to the TL feature space modeling. Consequently, they contain TL information on the feature distribution.

To perform these d TL clusterings, we use the MDL based Gaussian mixture modeling algorithm defined in section 3.2.2.

2.3.3.3 Complementarity of the representations

The MT feature space contains global information including the TL information. Moreover, for TL clustering, the separability of the different clusters is not as clear as for the MT case. However, TL analysis in contrast to the analysis of the highly dimensional MT feature space, allows a more detailed information extraction.

Consequently, as the interest is a time decomposition of the signal, one should associate these 2 different TL representations for a complete understanding of dynamic clusters.

2.3.4 Modeling the dynamic feature space

Our interest is now focused on modeling the signal represented in the dynamic feature space.

Each MT cluster has a particular behavior when observing its evolution into successive TL representations. For example, some MT clusters may share the same cluster at a given time and split or/and merge with other MT classes at another time. The problem is to quantify, at a given time, the similarity of these projected MT clusters with the goal of inferring spatio-temporal relations. Since our interest is particularly time locations and according to the previous remarks on the complementarity of both TL representations, we propose the following model of trajectories : the trajectories of the projected MT clusters $C^k_{MT_{t_i}}$ at the different times $\{t_i; i = 1, ..., d\}$ are formalized using collections of TL clusters $C^j_{t_i}$.

2.3.4.1 Inference of dynamic cluster

Based on these considerations, we define a model, noted as M_G, for the dynamic cluster trajectories. This model is a probabilistic distribution on the MT cluster collection $\boldsymbol{C}$ conditioned by a graph of trajectories $\boldsymbol{\mathcal{G}}$ constituted with TL clusters. Thus, we define the likelihood $p(\boldsymbol{C} \mid \boldsymbol{\mathcal{G}}, M_G)$ of a given a graph of trajectories. To express this distribution, we hereby introduce a few notations. We decompose the graph in a set of K_{MT} graphs of trajectories $\boldsymbol{\mathcal{G}} = \{\mathcal{G}_1, ..., \mathcal{G}_{K_{MT}}\}$ formed by TL clusters, where each graph $\mathcal{G}_k$ is associated to an MT cluster $C^k_{MT} \in \boldsymbol{C}$. By independence assumptions, we decompose this joint probability distribution $p(\boldsymbol{C} \mid \boldsymbol{\mathcal{G}}, M_G)$ into a simple product of probabilities $p(C^j_{t_i} \mid C^k_{MT_{t_i}})$. We refer the reader to section 6.2.1 of [19] for further details on this decomposition. To evaluate the latters, we map into probabilities the Kullback-Leibler divergence [16] noted as $D(C^k_{MT_{t_i}}, C^j_{t_i})$, which is an entropic measurement able to compare the two different TL distributions and learn about their similarity. Because of the relative interest for each of the MT class separately, a spatial constraint is introduced : this divergence measurement is weighted by the number of data points belonging to both of the corresponding MT and TL spatial classes.

Note that the clusters are characterized by multivariate Gaussian distributions and thus, the divergence calculation is performed analytically. The maximum of the likelihood probability (c.f. Eq.) can be obtained by using graphs $\mathcal{G}_k$ comprising all the TL clusters $C^j_{t_i}$. But as the objective is to infer only the most likely associations of MT clusters with TL clusters, we limit the graphs complexity by removing

associations with TL clusters which possess a probability $p(C_{t_i}^j \mid C_{MT}^k)$ below a given threshold μ. Thus, the graphs $\mathcal{G}_k$ maximizing the likelihood $p(\boldsymbol{C} \mid \boldsymbol{\mathcal{G}}, M_G)$ are simply those constituted with TL clusters for which probabilities $p(C_{t_i}^j \mid C_{MT}^k)$ are higher than μ.

Hence, using MT clusters, we infer graphs of trajectories of dynamic clusters which are composed of TL clusters and where the complexity of the graphs depends on a threshold parameter μ. These graphs constitute a model for the signal representation in the dynamic feature space.

2.3.4.2 A graph of dynamic cluster

The image time-series has been previously submitted to several processing levels. They result in a specific representation which is a graph modeling the trajectories of dynamic clusters. The chronology of the time-series and the irregular sampling information are stored in the graph. The trajectories information is condensed in the nodes and branches of the graph. Figure 5 summarizes the description of the graph characterizing the dynamic clusters.

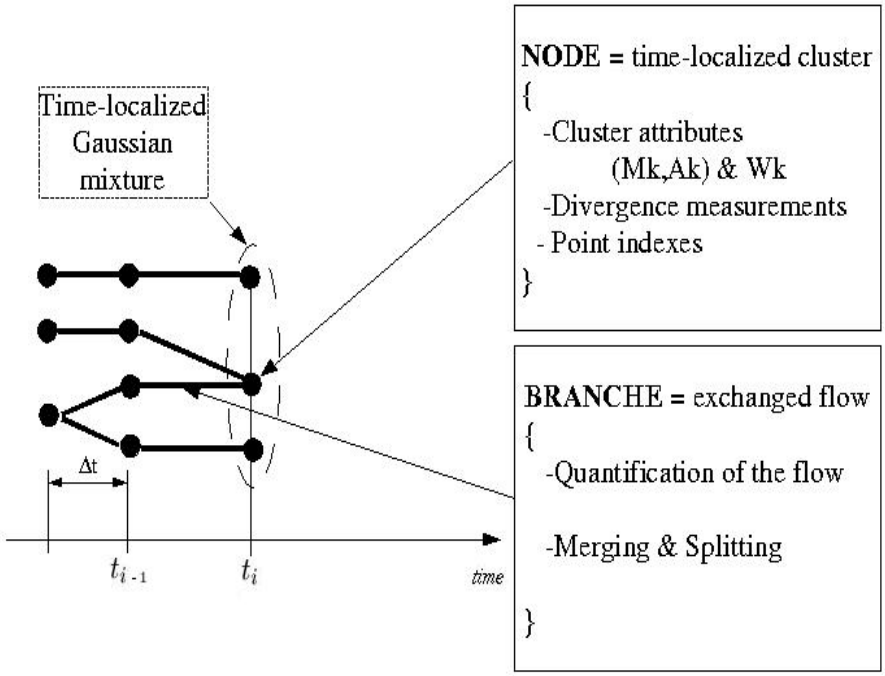


Figure 5 *: Description of the graph of dynamic cluster trajectories.*

- A node represents a TL cluster $C_{t_i}^j$ defining the j^{th} component of the Gaussian mixture at a given time t_i and is related to a collection of MT clusters by a set of probability measurements. The complete Gaussian mixture at a given time t_i is described by the entire set of TL clusters. Each MT cluster C_{MT}^k associated to the node $C_{t_i}^j$ is characterized by a pixel weight $card(C_{MT_{t_i}}^k(r) \cap C_{t_i}^j(r))$, a divergence measurement and TL Gaussian parameters. Moreover, each node regroups a set of indexed points in time and space represented in a TL class $C_{t_i}^j(r)$.

- The branches of the graph represent the MT cluster evolutions between two image samples. A branch, linking two consecutive TL clusters $C_{t_i}^j$ and $C_{t_{i-1}}^l$ which is related to a given MT cluster $\{C_{MT}^k\}$, is characterized by a time sampling interval Δt_i, a pixel flow and TL and MT Gaussian parameter evolutions. The flow of feature points exchanged between the TL cluster $C_{t_i}^j$ and $C_{t_{i-1}}^l$ is the number of pixels shared by the two TL spatial classes $C_{t_i}^j(r)$ and $C_{t_{i-1}}^l(r)$. Furthermore, because of the restriction to a given MT class, the flow is determined by the number of pixels shared by the previous TL spatial classes and the MT spatial class $C_{MT}^k(r)$. The quantization of flow of feature points enables us to evaluate the merging and the splitting of the dynamic clusters in time and in each feature dimension. These phenomena are simply related to the number of in going and out going branches associated to the nodes. The internal MT cluster changes between two consecutive times t_{i-1} and t_i can be quantified by mutual information. By using the projections of an MT cluster in two consecutive times t_{i-1} and t_i, mutual information between the two Gaussian projections can be measured using an analytical computation.

The graph characterizing the dynamic clusters $x_r(t,\theta)$ is a representation of the signal where the spatial variable r is hidden. However, spatial indexes related to each point in this feature space representation are accessible. Exploiting them permits us to generate representations in the image space. Indeed, we can associate to the K_{MT} different MT cluster trajectories, K_{MT} different representations in the spatio-temporal space. This space is formed by the spatial and temporal components r and t. We call these representations dynamic classifications. At each time $t_i \in [1,d]$, each dynamic classification is composed of a particular combination of TL classes $C_{t_i}^j(r)$. The TL classes of the dynamic classification related to the k^{th} MT class $C_{MT}^k(r)$, are those which correspond to TL clusters in the associated graph of trajectories $\mathcal{G}_k$.

These dynamic classifications contain spatio-temporal information missing in the dynamic feature space representation. Together, these representations describe objectively the feature evolution and the spatial evolution of the image time-series.

2.4 UNSUPERVISED INFERENCE OF A GRAPH OF DYNAMIC CLUSTER TRAJECTORIES

In this section we focus on a very important step in providing content-based query techniques: the interaction with the user and the flexible incorporation of user-specific interests. It constitutes the last level of the hierarchical information modeling. The semantic modeling detailed in this section was previously presented in [17]. The learning framework presents similarities with the one adopted by Schroder et al. [5].

Spatio-temporal processes, present at a given time and in a spatial window, can possess subjective user-specific semantics denoted by $\mathcal{A}_\nu$. A user may be interested in retrieving similar events and thus,

may want to know when and where similar spatio-temporal patterns occurred. Moreover, the inference of the graph $\mathcal{G}$ is a robust and unsupervised coding of SITS. And, as sub-graphs $\mathcal{G}_k$ contained in $\mathcal{G}$ are stochastic models for these spatio-temporal patterns, they can also possess user semantics. Therefore, based on this objective signal characterization, we are interested in learning semantics from users in order to achieve a semantic labeling of sub-graphs representing spatio-temporal patterns. Such a learning procedure could enable the recognition and the probabilistic retrieval of similar events.

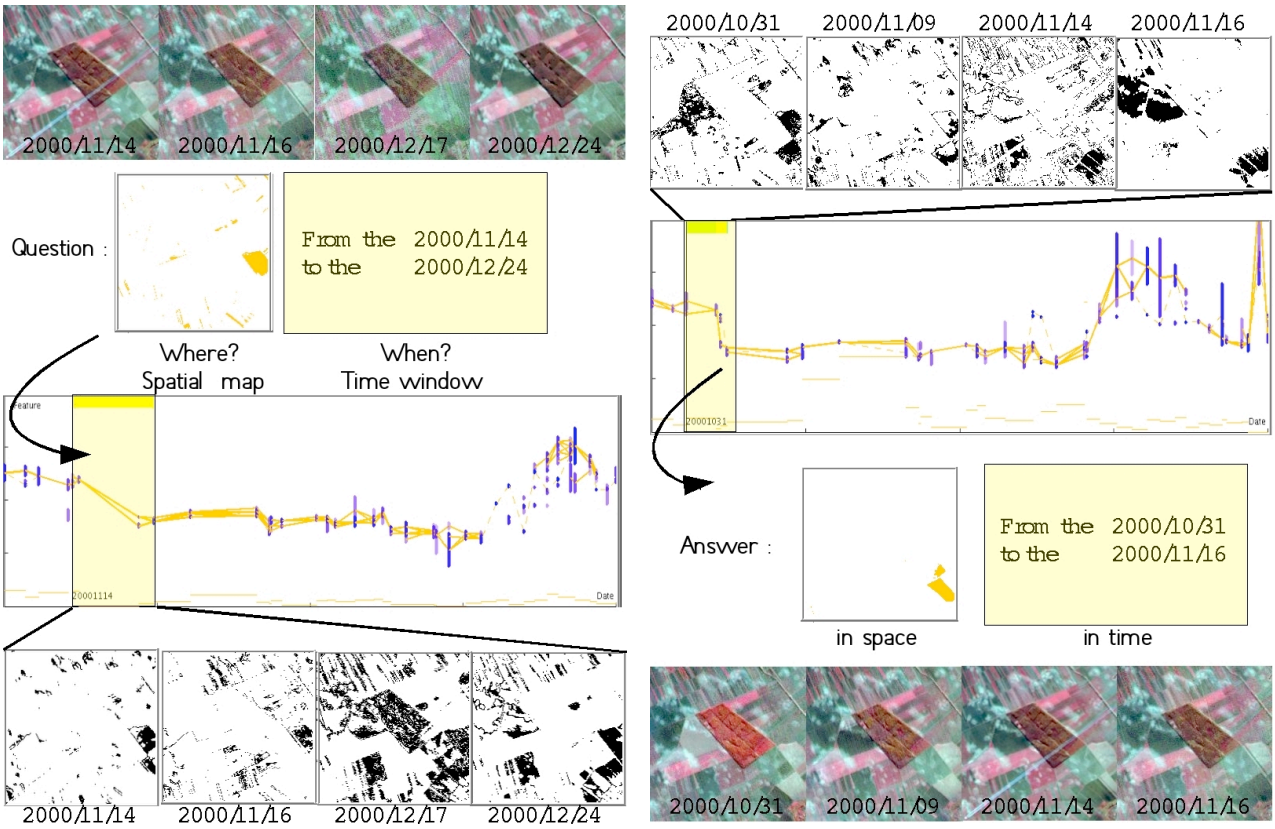


Figure 6 *: Interactions between a user and a graph-based learning system. The user transmits time-windows and a spatial masks to the system. They correspond to spatio-temporal patterns of interest or non-interest; based on the graph representations of these examples associated to their dynamic classifications, the system learns interactively user-specific semantics and retrieves, in time and in space, similar spatio-temporal structures***.**

In this perspective, we schematize in figure 6 the interactions between a user and a graph-based learning system : the user transmits to the system time-windows and spatial masks corresponding to spatio-temporal patterns of interest or non-interest; based on the graph representations associated to these positive and negative examples, the system learns interactively user-specific interests and retrieves a collection of spatio-temporal structures with similar semantics occurring in defined time-windows and spatial masks.

In order to define a model for user-specific semantics, a parametric similarity measure $S_\phi(\mathcal{G}_0, \mathcal{G}_k)$ between two sub-graphs $\mathcal{G}_0$ and $\mathcal{G}_k$ is employed [17]. This measure is an extension of the inexact matching algorithm proposed in [20]. The parameter vector ϕ weights the contribution of each type of

sub-graph features. A given parameter vector corresponds to a particular user-specific similarity and formalize a particular semantic.

By defining interactively a similarity, it is possible to link the subjective elements $\mathcal{A}_\nu$ representing the user semantics to the objective sub-graph features $\mathcal{G}_k$. For this we derive from the parametric similarity likelihood probabilities $p(\mathcal{G}_k \mid \mathcal{A}_\nu, M_S) \propto S_{\widehat{\phi}}(\widehat{\mathcal{G}}_0, \mathcal{G}_k)$, where $\widehat{\phi}$ is an estimated parameter vector and $\widehat{\mathcal{G}}_0$ is an estimated reference sub-graph. The estimation of this parameter vector is made interactively by updating the probabilities of a Bayesian network with user examples. More precisely, the probability update is performed indirectly by adjusting the hyper-parameters vector $\alpha = \{\alpha_i\}$ of a Dirichlet model, depending on the users' examples. For further details, we refer the reader to [17].

For notation simplification, the conditioning of the likelihood by a model M_S is omitted in the following. Based on the likelihood, using a Bayesian context enables the estimation of posterior probabilities $p(\mathcal{A}_\nu \mid \mathcal{G}_k)$ and thus, allows a semantic representation of the SITS content. Indeed, considering that a user provides positive or negative examples, corresponding to a positive $\mathcal{A}_\nu$ or negative $\neg\mathcal{A}_\nu$ semantics, two likelihood probabilities $p(\mathcal{G}_k \mid \mathcal{A}_\nu)$ and $p(\mathcal{G}_k \mid \neg\mathcal{A}_\nu)$ can be derived for each sub-graph. Moreover, graph priors can be obtained using the formula $p(\mathcal{G}_k) = \sum_\nu p(\mathcal{G}_k \mid \mathcal{A}_\nu) p(\mathcal{A}_\nu)$, where the summation is done over the positive and negative semantics. Thus, assuming a uniform prior on the semantics, the posterior probability $p(\mathcal{A}_\nu \mid \mathcal{G}_k)$ of the positive semantics is inferred using Bayes rule. For details on this Bayesian inference procedure, we refer the reader to chapter 7 of [19].

By interactive learning, user-specific semantic posterior probabilities $p(\mathcal{A}_\nu \mid \mathcal{G}_k)$ are obtained for each sub-graph $\mathcal{G}_k$. Therefore, a semantic labeling of sub-graphs is carried out which enables spatio-temporal reasoning and probabilistic retrieval of spatio-temporal structures in SITS.

3 EVALUATION OF THE SPATIO-TEMPORAL INFORMATION MINING SYSTEM

Author: Patrick Héas

3.1 INTRODUCTION

An exhaustive evaluation of an information mining system is not an easy task. It requires the development of specific tools able to measure the amount of information communicated between the data and the users [22]. Thus, the loss of information at each level of the model hierarchie has to be evaluated. In this document, we limit ourselves to validate the system by showing relevant examples of query results obtained by interactively learning several semantics.

Experiments presented in this document were performed using a SITS composed of SPOT multispectral images containing 2000x3000 pixels. The spatial resolution is 20 meters. The acquired scene is a rural area located in the East of Bucharest (Romania). The acquisition campaign was driven in order to provide remote sensing data for the *Data Assimilation for Agro-Modeling (ADAM)* project. The SITS was obtained by daily acquisition and by filtering out images presenting a cloud or a snow cover above the project test sites. This selection procedure resulted in 38 images irregularly sampled in time, which were acquired over a period of 286 days. The images were then made superposable and a radiative transfer model was applied to produce reflectance measurements. The ADAM project SITS is available on-line [1].

3.2 SPATIO-TEMPORAL REASONING

In order to demonstrate the capabilities of this spatio-temporal information system, which is able to exploit the information diversity contained in image time-series, we present in the following learning examples of various semantics, specific to dynamic rural scenes.

Ploughing of crops We begin by the search of spatio-temporal structures, associated to the ploughing of crops. We perform the learning with examples defined in a spatial window of 200x200 pixels and in a time-window constituted of the 38 samples. The searched structures are defined spatially by a MT class and temporally by a time-window of 4 samples. Note that prior knowledge is already introduced at this level by chosen the size of the time window including the structures. Candidates which posses different time scales are thus excluded from the search. By interactive learning of the semantic, posterior probabilities of structures are inferred and, by maximization of these probabilities, structures are retrieved in space and in time. The figure 7 present the retrieved structures possessing the highest posterior probabilities. The learning has enabled to retrieved the majority of the ploughing phenomena occurring in the data. Nevertheless, we note a few false detections. We then extend the search to a wider spatial window of size 800x800 pixels. The figure 8 presents the retrieved structures possessing the highest posterior probabilities and the associated dynamic classifications. As expected, we retrieve structures with similar semantic. Because the number of ploughing phenomena increases with the image size, we characterize better this semantic and false alarm decrease. Note that the visualization of the dynamic classification enables us to follow the spatial evolution, what is particularly interesting when the image size becomes important.

Note that the wider the time-window containing the structure is, the more they will be characterized and their semantics will be differentiated. Thus, the ploughing phenomena occur in a quite short time-period. Moreover, these phenomena possess variable characteristics, depending on the crop nature. Consequently, the differentiation of a positive semantic class from a negative semantic class is difficult..

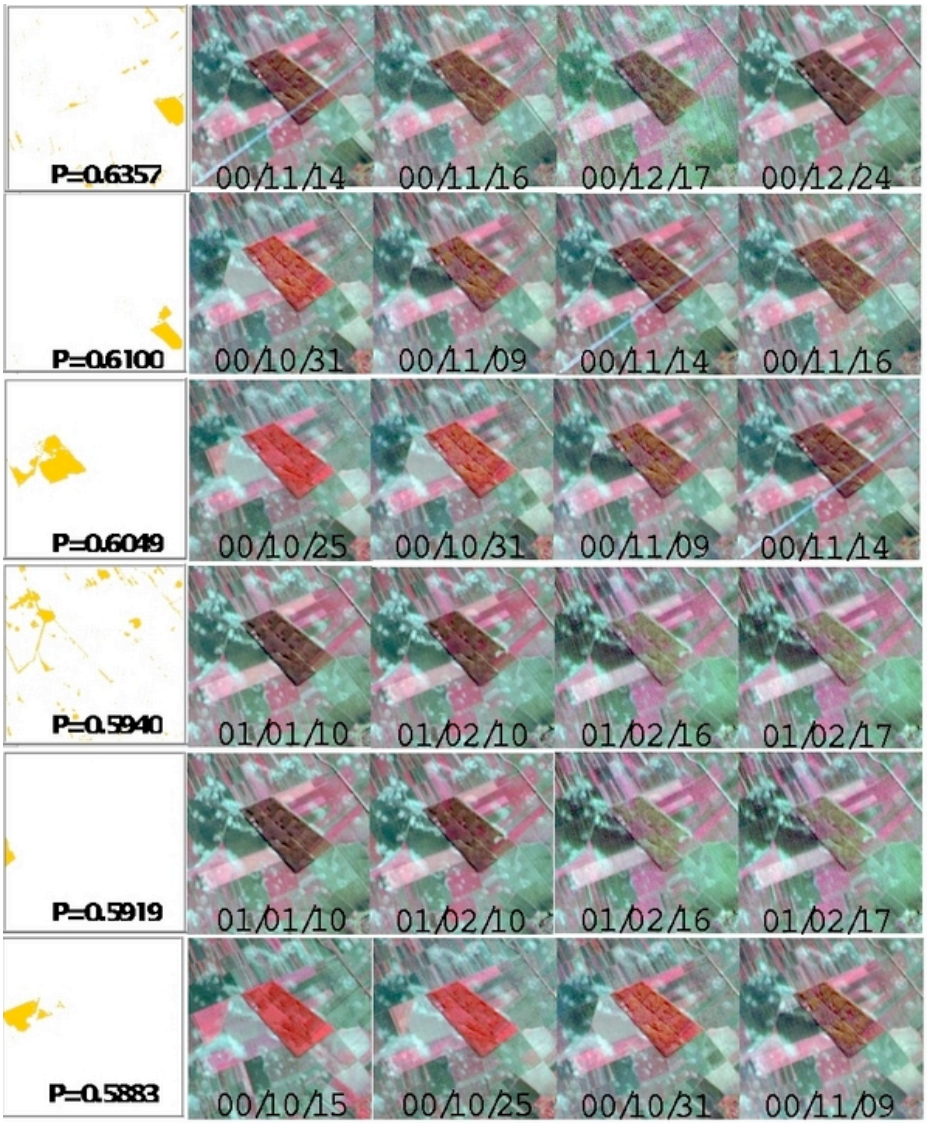


Figure 7 *: Spatio-temporal structures associated to the ploughing of crops, retrieved by interactive learning. The spatiotemporal structures are searched in a spatial window of 200x200 pixels and in a temporal window constituted of 38 samples. The most likely structures, defined spatially by a MT class and temporally by a time-window constituted of 4 samples, are retrieved in space and time. The figure presents the most likely spatio-temporal structures ranked, from top to bottom, according to their posterior probabilities P . Each line presents a retrieved spatio-temporal structure, i.e. a retrieved MT class (left) and a time window (given by the dates displayed in the first and last image of the line), and the image time-serie (right) where the spatio-temporal structure appear.*

Thus, we limit ourselves to the positive examples. Therefore, posterior probabilities are not very high and there is a moderate discrimination of this semantic

This example demonstrate the system capability to learn very accurate semantics, that is the ploughing after the harvest of a wheat crop. Moreover, in addition to the probabilistic retrieval of structures, the system enables the user to understand the spatial evolution of the ploughing phenomena via the visual analysis of the dynamic classification.

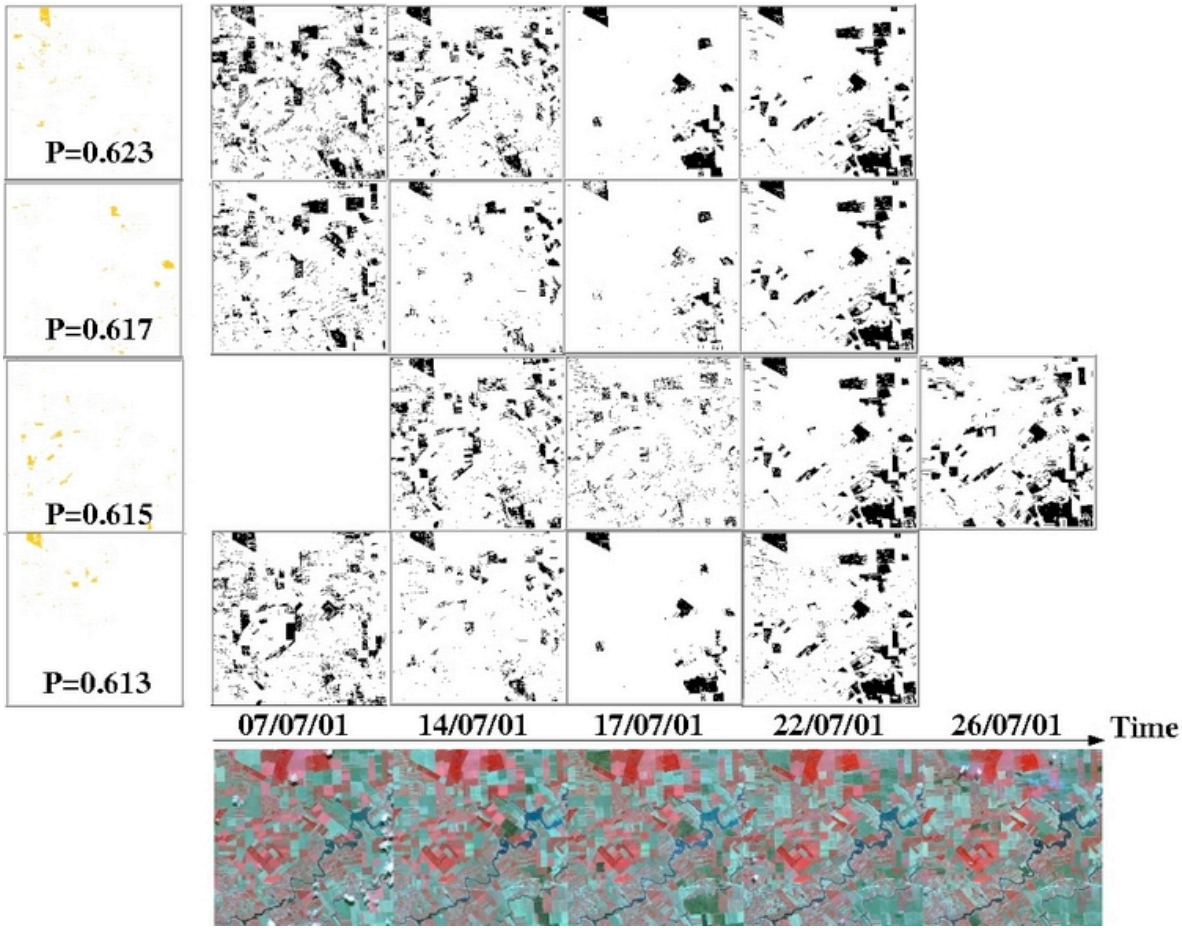


Figure 8 *: Spatio-temporal structures associated to the ploughing of crops, retrieved by interactive learning. The spatiotemporal structures are searched in a spatial window of 800x800 pixels and in a temporal window constituted of 15 samples. The most likely structures, defined spatially by a MT class and temporally by a time-window constituted of 4 samples, are retrieved in space and time. The figure presents the most likely spatio-temporal structures ranked, from top to bottom, according to their posterior probabilities P. Each line presents a retrieved MT class (left) and its associated dynamic classification (right). The time window in which is defined the retrieved structure is given by the temporal location of the first and last image of the line. The image time-serie, containing all the retrieved spatio-temporal structures, is presented in the bottom of the figure.*

Maturation of crops. We then train the system to retrieve spatio-temporal structures associated to the maturation of crops. The structures are searched in a spatial window of 200x200 pixels and in a time-window constituted of the 38 samples. As the maturation phenomena occur on a rather long time period, we search structures included in a time window of 12 samples. We communicate a few positive examples which are enough for the discovery of similar structures. Contrary to the previous ploughing semantic, the learning of a negative semantic is performed with very few examples. This learning facility is likely to be induced to the important width of the considered time window. We thus obtain a relevant learning of both positive and negative semantics. Consequently, the retrieved structures are associated to high posterior probabilities. The results of the search in space and time is presented in figure ç. The spatio-temporal structures are represented together with sub-sampled time-series (only three images have been selected from the 12 samples of the time-serie). Note that the crop evolutions

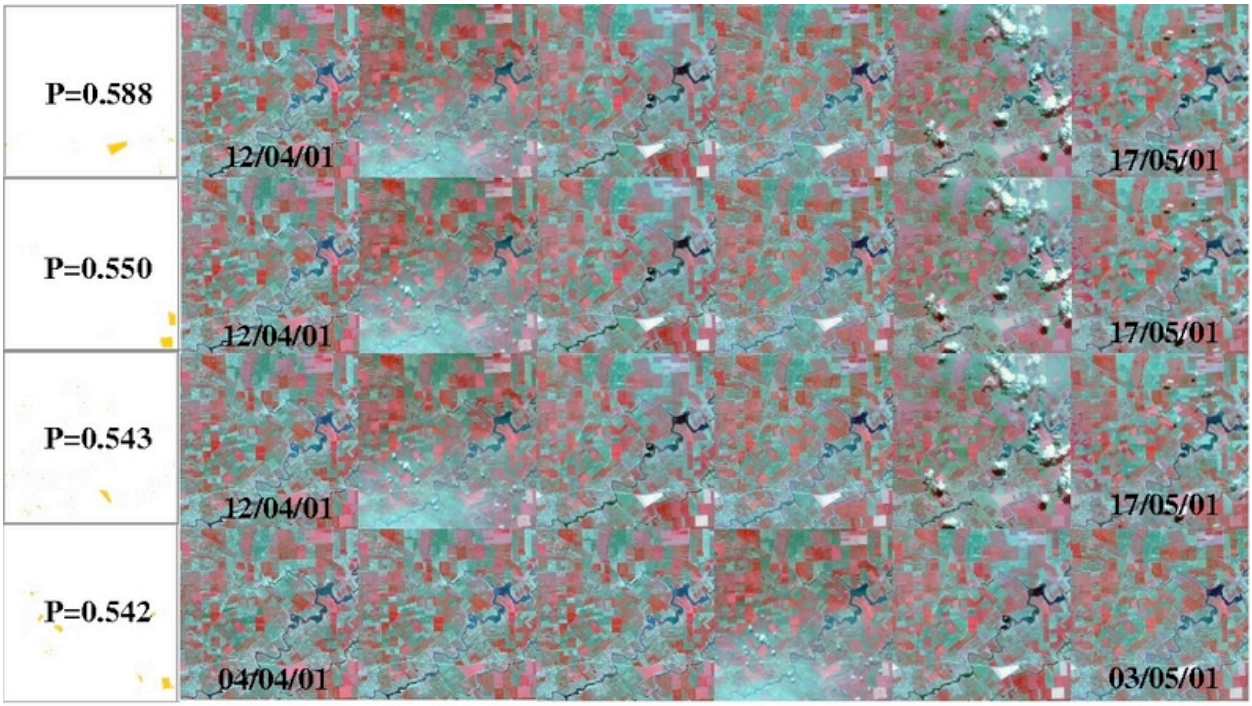


Figure 9 *: Spatio-temporal structures, associated to the maturation of crops, retrieved by interactive learning. The spatiotemporal structures are searched in a spatial window of 200x200 pixels and in a temporal window constituted of 38 samples. The most likely structures, defined spatially by a MT class and temporally by a time-window constituted of 12 samples, are retrieved in space and time. The figure presents the most likely spatio-temporal structures ranked, from top to bottom, according to their posterior probabilities P. Each line presents a retrieved spatio-temporal structure, i.e. a retrieved MT class (left) and a time window (given by the dates displayed in the first and last image of the line), and the image time-serie (right) where the spatio-temporal structure appear.*

during the training, while the evolutions associated to lower probabilities correspondent to the maturation of different cultures.

This example highlights the system capability to learn a very general semantic which is associated to quite different structures. Indeed, the maturation semantic can be associated to the specific culture of soja, but can also be associated to the increasing of a forest biomass.

Evolution and blooming of colza. We focus now on the training of a semantic associated to the blooming of colza. The colza blooming period is quite short, a few weeks, and occurs in spring. Consequently, the searched structures are included in time window constituted of of 6 samples comprised between march and june. A single example communicated to the system is enough to retrieve similar phenomena in a spatial window of 800x800 pixels. The figure 4 presents the structures possessing the highest probabilities. We note that because any negative examples have been communicated to the system, the probabilities remain low. The presence of an aerosol is interpreted as a blooming phenomenon.

If we communicate negative examples to the system then, the semantic class differentiate slightly. The figure 10 presents colza crops, retrieved and clearly differentiated from other cultures, retrieved in a window containing 200x200 pixels and in a time window of 38 samples.

This result is congruent with the previous remarks and strengthen our intuition : the wider the time-window containing the structure, the more differentiate the semantic.

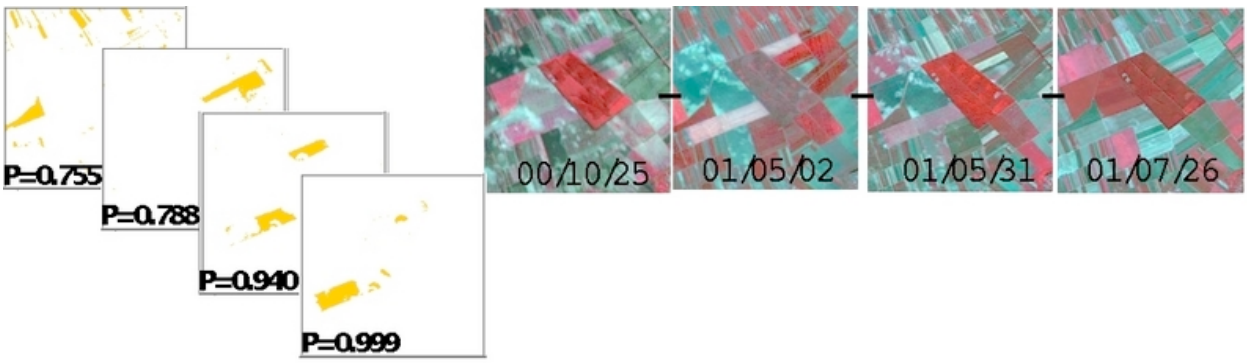
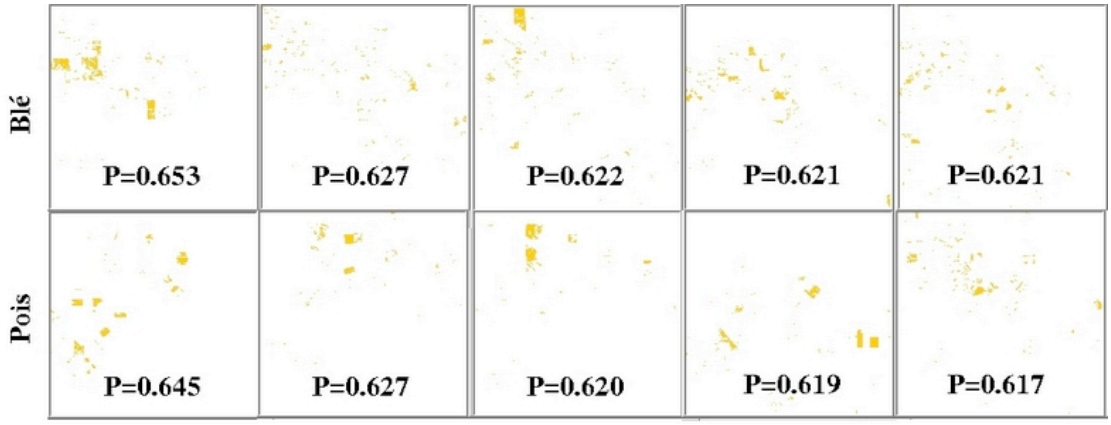


Figure 10 : *Spatio-temporal structures, associated to the blooming of colza crop, retrieved par interactive learning with a single positive example. The spatio-temporal structures are searched in a spatial window of 800x800 pixels and in a temporal window comprising samples acquired between march and june 2001. The most likely structures, defined spatially by a MT class and temporally by a time-window constituted of 6 samples, are retrieved in space and time. The figure presents the most likely spatio-temporal structures ranked, from top to bottom, according to their posterior probabilities P . Each line presents a MT class (left) and its associated time window (given by the dates displayed in the first and last image of the line).*

Cultural practice for wheat and pea. Finally, we research spatio-temporal structures associated to crops submitted to some cultural practice. We thus focus on identical annual evolutions and more precisely to the wheat cycle : in autumn, the crop is ploughed then sowed; the culture vegetate during winter; during spring, the cultures grow until maturation; at the end of summer, the wheat is finally harvested. We also consider the evolution of pea culture : the evolution is characterized by the leaves development and ramification during spring, a blooming in the beginning of june and a harvest in august. We search spatio-temporal structures in a spatial window of 800x800 pixels. In this spatial window, a single positive example is chosen, defined spatially by a MT class and temporally by a maximal time-window constituted of 38 samples. We retrieve so crops of same nature which have been submitted to identical changes. For instance, we will be able to differentiate crops of similar nature which have been harvested in different periods.

The most likely structures retrieved after that we provided a particular wheat or pea crop to the system, are presented in figure 11. First of all, we observe a sparse spatial repartition of the structure associated to these semantics. Then, in order to understand why these structures have been retrieved with the highest posterior probabilities, we need to inspect carefully the structure temporal evolutions visible in the image time-serie. This visual inspection enables us to conclude that similar cultural practice characterize the retrieved crops. Consequently, this example demonstrate the system capability to recognize complex phenomena, spread in space and marked in time by similar events. Obtaining similar results by only inspecting visually the data would require a lot of man power.



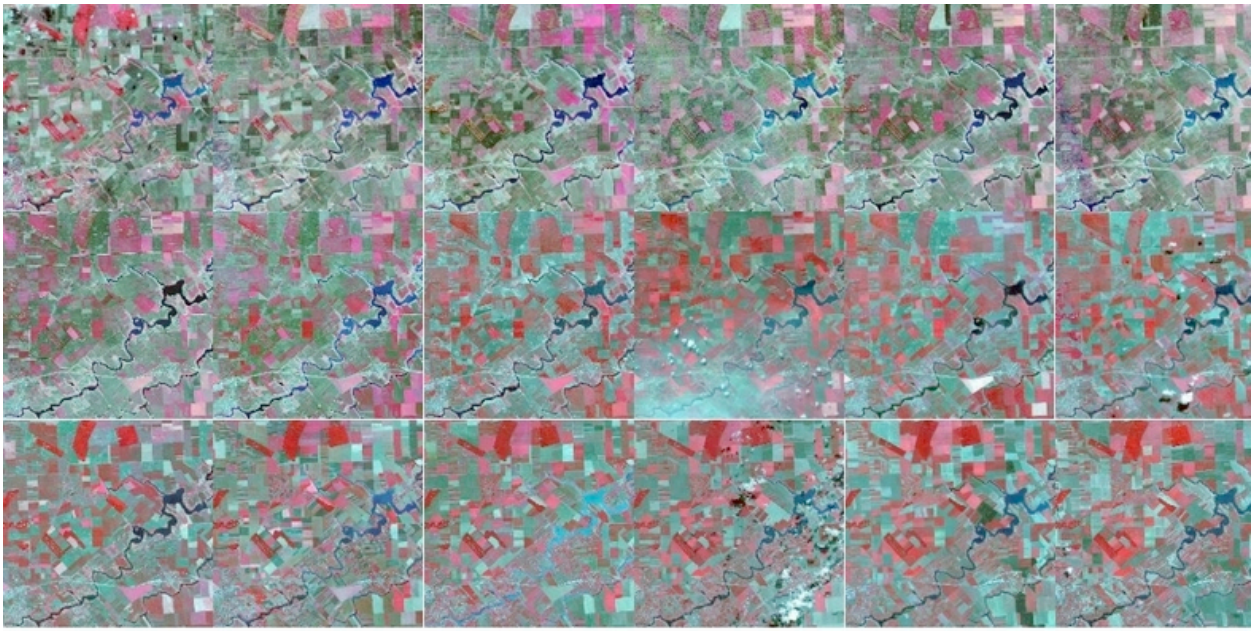


Figure 11 *: Spatio-temporal structures, associated to some cultural practice for wheat or pea, retrieved par interactive learning with a single positive example. The spatio-temporal structures are searched in a spatial window of 800x800 pixels and in a temporal window comprising 38 samples. The most likely structures, defined spatially by a MT class and temporally by a maximal time-window constituted of 38 samples, are retrieved in space. A single example of evolution of wheat or pea crops, enables the system to discover crops of same nature submitted to the same cultural practice (same period of harvest, ploughing, etc). The figure presents on two lines, corresponding to the 2 different learnings, the most likely spatio-temporal structures (above). On each line, the retrieved MT classes are ranked, from left to right, according to their posterior probabilities P . The image time-serie, temporally sub-sampled, is presented on 3 lines (below). The latter contains all the retrieved spatio-temporal structures.*

3.3 CONCLUSION

Through these semantic labeling examples, the power of the system has been highlighted. Indeed, the variety of the information content of satellite image time-series has been exploited. The modeled semantic can be associated to very specific phenomena but also to more general evolutions. In addition, these examples have demonstrated the system performance to search structures in space and in time and to understand them. Nevertheless, we noted difficulties to learn negative semantics, i.e. to discriminate positive semantics, when considering too narrow time-windows to characterize structures. These difficulties are probably induced by the semantic model limitations [19].

4 THE SPATIO-TEMPORAL INFORMATION MINING SYSTEM AND ITS IMPLEMENTATION

Authors: Patrick Héas / Alain Giros

This section describes the general architecture of the spatio-temporal information mining system. The latter is based on the theoretical concept which has been introduced in section 2 and evaluated in section 3. As exposed previously, this concept enables us to link the interest of a user to specific spatio-temporal structures.

The system is decomposed in two parts : an unsupervised modeling of spatio-temporal structures resulting in a graph, and an interactive learning procedure based on graphs which leads to the semantic labeling of spatio-temporal structures. Fig.1 and Fig.2 present the general architecture of the unsupervised learning system and of the interactive learning system.

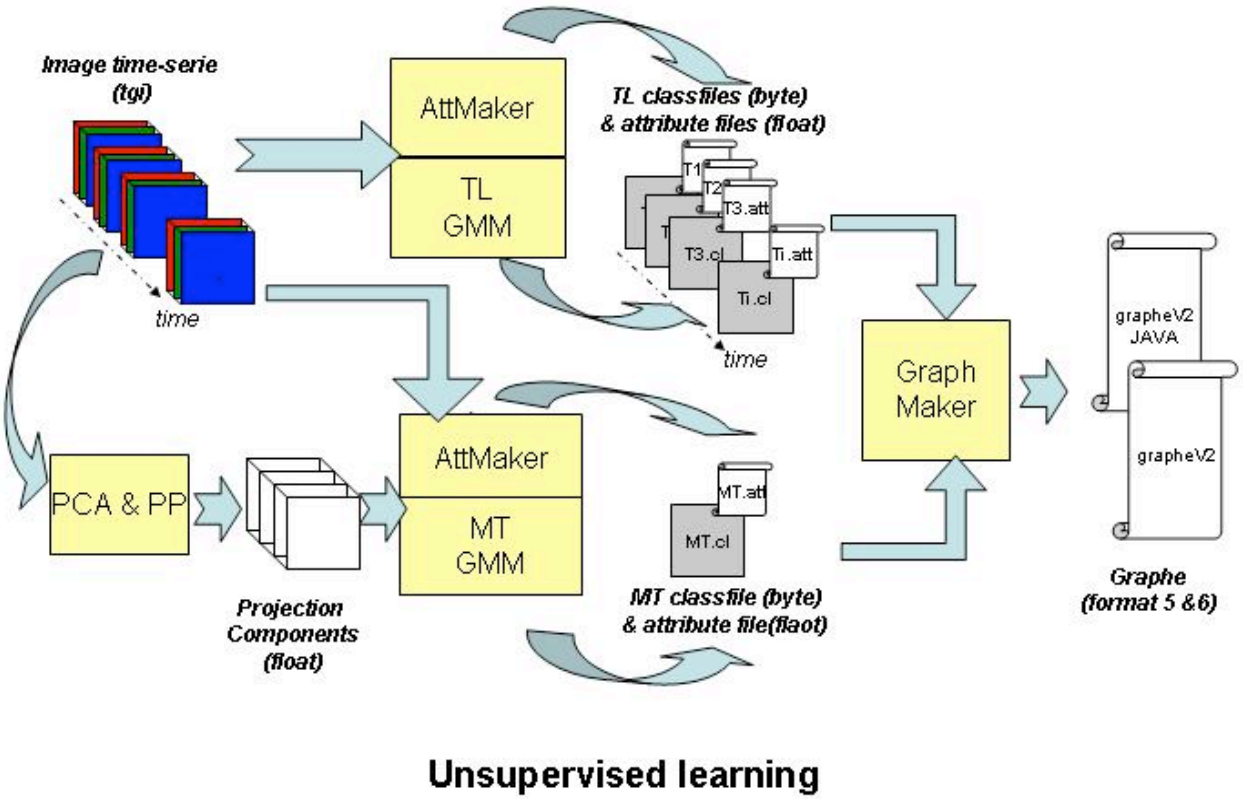


Figure 12. *Architecture of the graph unsupervised learning system*

Unsupervised learning architecture. Two complementary representations are first induced by placing the image time series features (spectral, textural;) in the Time-Localized (TL) feature spaces and in the MultiTemporal (MT) feature space. On the latter, we apply the dimension reduction algorithm "PCA &PP" to produce a condensed description of the MT feature space by extracting several interesting projection components. We then use the "TL GMM" and "MT GMM" algorithms to model the different feature spaces by Gaussian Mixture Modeling (GMM). It results in classification maps named with the extension ".cl". For each produced classification map, we apply the "AttMaker" algorithm to generate classifications attribute files named with the extension ".att" This algorithm takes as input files a classification map with its corresponding image components : for the TL representation, each map correspond to a given multispectral image, while for the MT representation, the classification map correspond to all the images of the time-serie. Then, the TL/MT classfiles and attribute files are used as an input of the "GraphMaker" algorithm which infers the graph of cluster trajectories. The algorithm produces graph files at various format (e.g. Format 5, Format 6). This programs concludes the unsupervised learning part.

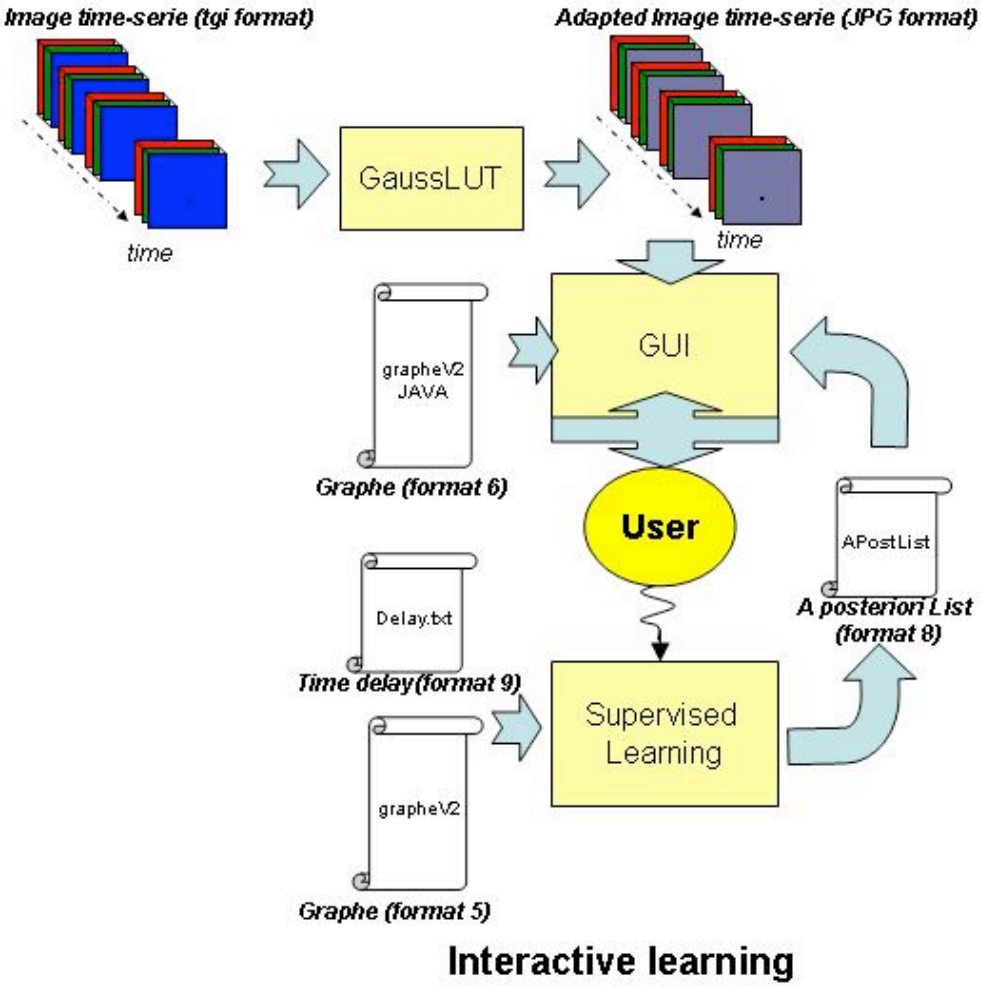


Figure 13. *Architecture of the interactive learning system*

Interactive learning architecture. The interactive learning is performed by the "SupervisedLearning" algorithm and employs a Graphical User Interface abbreviated as GUI.

The "SupervisedLearning" algorithm takes as inputs the graph (at format 5) and an extra text file called "Delay.txt" (format 9). Semantic labellings are enabled by user-provided positive and negative examples. The user uses the GUI to find the spatio-temporal locations of his structure of interest. These locations are transmitted to the learning system by the standard input (keyboard) . The "SupervisedLearning" algorithm generates after each learning step a text file called "ApostList"which is an a posteriori list together with relevance feedback parameters. The latter is used as an input by the GUI in order to enable the user to visualize the current semantic labeling.

The GUI uses as input the graph (at format 6) and an adapted image time-serie in JPG format. We remark here that the time delay between consecutive images of the time series is introduced directly in the applet code (c.f. Part VII). The adapted image time series is generated by applying the "GaussLUT" algorithm on the original data and converting the raw images in JPG quiklooks using Unix/Linux commands. By simple clicks, the user can navigate in the different data representation and view the current semantic labeling provided by the "ApostList" input file.

We now detail each module of the unsupervised and supervised learning chains.

4.1 PCA &PP : MULTIDIMENSIONAL ANALYSIS AND DIMENSION REDUCTION BY PRINCIPAL COMPONENT ANALYSIS AND PROJECTION PURSUIT

This program : first analysis a multidimensional feature space composed by several images bands by Principal Component Analysis (PCA). It produces as an output the principal components. and second analysis a reduced multidimensional feature space composed by only principal components representing a signal energy percentage (fixed by the user) by Projection Pursuit (PP). It produces as an output the extracted projections.

For an overview on the used models and inferences, we refer the reader to the concept description. For further details, we refer the reader to section 6.1.2 of [19].

a)Activation syntax

i. The program is in the following directory
../DoxygeneCode/Learning/NonSupervisedLearning/DimReduc
The name of the compiled program implementing this module has the following syntax : *'PCA_PP'+terminology*
where the string terminology is either '*_SI'* or'*_F'*, specifying that the input data is either in short integers or in floats.

ii. The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space:

→Required parameters :
- arg[1] Text file name containing the names of input image bands. The number of lines of this file must be the total number of bands
- arg[2] Text file name containing the names of output projections created by PP

- arg[3] Text file name containing the names of output components created by PCA
- arg[4] Number of lines of the images
- arg[5] Number of columns of the images

→Optional parameters fixing dimension reduction factors :

- arg[6] Signal energy percentage to preserve with PCA (default is 90%)
- arg[7] Quantile : index projection limit to stop the search (default is 0.7)

→Optional parameters for selecting a spatial window in the 'image steak' :

- arg[8] Column offset when reading the image file (default is 0)
- arg[9] Line offset when reading the image file (default is 0)
- arg[10] Number of columns of the image file(default is arg[5])

→Optional parameters for tuning the PP optimization algorithm :

- arg[11] Number of rand_departures (default is 5)
- arg[12] Maximum number of iterations (default is 50)
- arg[13] Precision of the convergence (default is 1e-3)
- arg[14] Factor decreasing the scale of the search at each iteration (default is 2)

iii.Activation syntax example :

Move to the Example directory

cd *../DoxygeneCode/Learning/NonSupervisedLearning/DimReduc*

Run the script *./example.sh* which contains the following line :

./PCA_PP_SI inputData.txt outPP.txt outPCA.txt 100 100 95 0.7 400 200 3000

b)Input & output formats

- The input image bands are stored in individual files . Their format depends on the compiled program terminology.

  To create the input text file , the names of each image band of the multidimensional image must be listed one after the other in consecutive lines
- The output image components and projections are always saved individually in float format.

  To create the output text file containing the PCA/PP components/projections , the names of the components/projections must be listed one after the other in consecutive lines

c)Usage recommendation:

The analysis by PP requires quite a lot of computation power when dealing with big images or/and a high dimensionality. However, because the PCA analysis is an analytical calculation it is performed very fast. Note that the PCA components are created and saved on the disk before going into the PP procedure

4.2 TL & MT GMM: GAUSSIAN MIXTURE MODELING OF MULTIDIMENSIONAL SPACES BY THE MDL PRINCIPLE

This is the program for inferring by MDL using a data 2-part coding, a Gaussian mixture model of unknown complexity (parameters and number of Gaussians).

Modeling procedures are chained. for clustering consecutive TL feature spaces. The initialization is done using the Gaussian parameters inferred for the previous modeling. Furthermore, the algorithm checks that there is a sufficient number of Gaussians (fixed by the user) after an initialization, and adds new Gaussian if necessary. For the MT feature space a independent modeling procedure is performed.

For an overview on the used models and inferences, we refer the reader to the concept description. For further details, we refer the reader to section 6.1.3 & section 6.1.4 of [19].

a)Activation syntax

i.The program is in the following directory

../DoxygeneCode/Learning/NonSupervisedLearning/GM_Modeling

The name of the compiled program implementing this module has the following syntax :*'MDLGMM'+terminology*

where the string terminology is either *'_SI'* or *'_F'*, specifying that the input data is either in short integers or in floats.

ii.The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space:

→Required parameters :

- arg[1] Text file name containing the names of input multiband-images : The number of lines of this file must be (numberImagesXnumberBands)
- arg[2] Text file name of output image classfiles
- arg[3] Number of bands of each image
- arg[4] Number of lines of the images
- arg[5] Number of columns of the images

→Optional parameters for selecting a spatial window in the 'image steak' :

- arg[6] Line offset when reading the image file (default is 0)
- arg[7] Column offset when reading the image file (default is 0)
- arg[8] Number of columns of the image file(default is arg[5])

→Optional parameters for spatial sub-sampling :

- arg[9] factor to reduce each line (default = 1)
- arg[10] factor to reduce each column (default = 1)

→Optional parameters for fixing the initial number of Gaussians :

- arg[11] Minimum number of Gaussians in the mixture for initialization (default is a function of arg[3],arg[4],arg[5].

ii.Activation syntax example :

Move to the directory

cd ../DoxygeneCode/Learning/NonSupervisedLearning/GM_Modeling

Run the the script *./example.sh* which contains the following lines :

./MDLGMM_SI in.txt out.txt 3 100 100 200 400 3000

./MDLGMM_F inMT.txt outMT.txt 2 100 100

b)Input & output formats

- Each input image band is stored individually in a file. Their format depends on the compiled program terminology.

Too create the input text file containing the multispectral image band names, the names of each image band of each multidimensional image must be listed one after the other in consecutive lines.

(e.g. : ../image1_band1

../image1_band2

../image2_band1

../image1_band2)

- The output image components are always saved individually in float format.

To create the output text file containing the classfiles names , the names of the classfiles must be listed one after the other in consecutive lines.

c)Usage recommendations :

Be aware that for a fixed number of samples, the higher the dimensionality is, the less the space is be populated. So, if the initial number of component (arg[11]) is too high, it might happen that the initialization fails. For special case, it might also happen, that the initial Gaussian components spread according to a mean and variance criterion fail to produce a good initial Gaussian Mixture and the algorithm reveal a false Gaussian Mixture of very few components . This situation was observed for instance when data possessed strong outliers

4.3 ATTMAKER : CREATION OF THE CLASSIFICATION ATTRIBUTE FILES

Program creating an attribute file associated to a classification map and a stack of image. The TL classifications together with their corresponding multispectral image constitute the algorithm inputs.

The MT classification is associated to the stack of images of entire time-serie.

.

a) Activation syntax

i.The program is in the following directory

../DoxygeneCode/Learning/NonSupervisedLearning/MakeClusterAtt

The name of the compiled program implementing this module has the following syntax :*'MakeClusterAtt'+terminology*

where the string terminology is either *'_SI'* or *'_F'*, specifying that the input data is either in short integers or in floats.

ii.The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space:

→Required parameters :

- arg[1] Text file name containing an input multidimensional image : the image bands are listed one after the other;
- arg[2] Name of the input classfile
- arg[3] Name of the output classification attribute file
- arg[4] Number of lines of the images
- arg[5] Number of columns of the images

→Optional parameters for selecting a spatial window in the 'image steak' :

- arg[6] Line offset when reading the image file (default is 0)
- arg[7] Column offset when reading the image file (default is 0)
- arg[8] Number of columns of the image file(default is arg[5])

→Optional parameters for spatial sub-sampling :

- arg[9] factor to reduce each line (default = 1)
- arg[10] factor to reduce each column (default = 1)

iii.Activation syntax example

Move to the directory

cd ../DoxygeneCode/Learning/NonSupervisedLearning/MakeClusterAtt

Run the script ./example.sh which contains the following lines :

ls ../data/20001015.tg.* > inIm.txt*

./MakeClusterAtt_SI inIm.txt ../data/20001015.cl ../data/20001015.att 100 100 400 200 3000

ls ../data/20001025.tg.* > inIm.txt*

./MakeClusterAtt_SI inIm.txt ../data/20001025.cl ../data/20001025.att 100 100 400 200 3000

ls ../data/20001031.tg.* > inIm.txt*

./MakeClusterAtt_SI inIm.txt ../data/20001031.cl ../data/20001031.att 100 100 400 200 3000

ls ../data/20001015.tg.* ../data/20001025.tg*.* ../data/20001031.tg*.* > inIm.txt*

./MakeClusterAtt_SI inIm.txt ../data/MT.cl ../data/MT.att 100 100 400 200 3000

b) Input & output formats

- The input image bands are stored in individual files . Their format depends on the compiled program terminology.

  To create the input text file , the names of each image band of the multidimensional image must be listed one after the other in consecutive lines
- The output classification attribute files are in text files at format defined in Format 4) (c.f appendix A)

4.4 GRAPHMAKER : INFERENCE OF A GRAPH OF DYNAMIC CLUSTER TRAJECTORIES

Program for the inference of a graph of dynamic cluster trajectories. For an overview on the used models and inferences, we refer the reader to the concept description. For further details, we refer the reader to section 6.2 of [19].

a)Activation syntax

i)The program is in the following directory
../DoxygeneCode/Learning/NonSupervisedLearning/GraphInference

The name of the compiled program implementing this module is the following :*GraphMaker*

ii)The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space:The program is in the following directory

→Required parameters :
- arg[1] Text file name of input TL classfiles and attribute files. The classfiles and attribute files are listed one after the other; The number of lines of this text file must be equal to 2 X numberOfClassfiles.
- arg[2] Text file name of input MT classfile and attribute file? The classfile and attribute file are listed one after the other; The number of lines of this text file must be equal to 2
- arg[3] Dimensionality of the TL feature space
- arg[4] Number of lines of the images
- arg[5] Number of columns of the images

→Optional parameters :
- arg[6] Format of the Graph out puts

  1:standard, 2: adapted to the actual interactive learning and GUI (default = 1)
- arg[7] Resolution factor for the divergence integration (default = 10)
- arg[8](0= mean/1= divergence). If "0" the divergence calculation between consecutive TL clusters for edge characterization is replaced by a simple mean difference (default =0)

iii)Activation syntax example :

Move to the directory

cd ../DoxygeneCode/Learning/ NonSupervisedLearning/GraphInference

Run the script *./example.sh* which contains the following lines :

./GraphMaker TL.txt MT.txt 3 100 100 2 10. 0

mv grapheV2 ../../../data/*

b) Input & output formats

- The input TL and MT classfiles are in bytes while the TL and MT classification attribute file are at format defined in Format 4) (c.f appendix A).

  To create the input text file with the TL classfile and attribute names , the names of each TL classfile alternated with the name of the classification attribute file must be listed one after the other in consecutive lines. The number of lines of this text file must be : numberClassfiles X 2.

  To create the input text file with the MT classfile and attribute , the names of the MT classfile followed by the name of the classification attribute file must be listed on two lines.
- The output graphs are either in the :standard format or defined in Format 5) and 6), where Format 5) is adapted to the actual supervised learning implementation and Format 6) is adapted to the actual GUI (c.f appendix A).

c) Usage recommendations:

The integration needed when the edge characterization is done with Kullback-Leibler divergence (arg[8]=1) may require a long calculation time. As the actual learning algorithm runs only with mean calculation, it is for the format 1 is for the moment not needed.

4.5 SUPERVISEDLEARNING : INTERACTIVE LEARNING

Program for learning interactively user-specific semantics attached to sub-graphs.

For an overview on the used models and inferences, we refer the reader to the concept description. For further details, we refer the reader to the chapter 7 of [19]

a) Activation syntax and Man-Machine dialogue

i. The program is in the following directory

../DoxygeneCode/Learning/SupervisedLearning

The name of the compiled program implementing this module is the following : *SupLearnPara*

ii. The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space:

→Required parameters :

- arg[1] Graph file name at format 5 (c.f appendix A)
- arg[2] Text file name containing, on a single line and separated by a white space, the time delay between the different time samples specific to the given graph.

iii. Activation syntax example :

Move to the directory

cd ../DoxygeneCode/Learning/SupervisedLearning

Run the script *./example.sh* which contains the following line :

./SupLearnPara ../../../data/grapheV2 ../ ../../data/TimeDelays.txt

iv. Man-Machine dialogue

1) User initializes the learning procedure :

Once the program is launch the system ask the user to transmit him several parameters to initialize the learning procedure using the standard input

Using the GUI (c.f. section 4.6) the user visualizes the different information representation, which enables him to respond the following questions :

1. *number of samples of the time-window?*

   That is the time-window in which spatio-temporal structures will be defined
2. *Minimum sample time of the graph for searching other sub-graphs?*

   That is the time sample index corresponding to the time lower bound above which the system will search sub-graphs of the similar semantic
3. *Maximum sample time of the graph for searching other sub-graphs?*

   That is the time sample index corresponding to the time higher bound bellow which the system will search sub-graphs of similar semantic
4. *Divergence threshold?*

   That is the the weighted divergence measurement (not normalized to probabilities -> to be done!!) tuning the graph complexity. Should be fixed according to the visualization of the graph with the GUI. A standard value is 30. If this value is set to 0, the system understands that the user is interested in only MT most likely trajectories.
5. *First sample of the sub-graph of interest?*

   That is the time sample index corresponding to the lower bound of the temporal window in which is defined the spatio-temporal structure example
6. *Index of the MT Class of interest?*

   That is the MT class index comprising the spatio-temporal structure example
7. *Do you want to fix a priori parameters for the similarity model ('y'/'n')?*

   If prior knowledge on the parameters exist respond 'y' and then follow the instruction. Otherwise respond 'n'.
8. *Do you want to fix some specific parameters for the search ('y'/'n')?*

For experimented user responding 'y' enables to fix the number of branches to consider in the search tree for the similarity model optimization For novice , please respond 'n'

After fixing these initialization parameters, the system creates 1) the sorted a posteriori list of sub-graphs and 2) relevance feedback measurements (c.f. third section).

2) Learning by communication of user's examples to the system

To enhance the learning, visualizing the spatio-temporal structures through the GUI (c.f. VI), the user continue providing to the system locations of structure examples : :

1.LOCATION OF SUB-GRAPH OF INTEREST (1) OR OF NON-INTEREST (-1)?

User must respond '-1' if he wants to provide a negative example or '1' is he wants to provide a positive one

2.First sample of the sub-graph of interest or non-interest?

That is the time sample index corresponding to the lower bound of the temporal window in which is defined the spatio-temporal structure example

3.Index of the MT Class of interest or non-interest?

That is the MT class index comprising the spatio-temporal structure example

4.do you want to keep the actual reference subgraph: (time sample=X, Index MT class=X)? (yes='1'/no='0')#?

After giving the user the spatio-temporal location of the most likely sub-graph of the a posteriori list, the system asks if he should update the reference subgraph:

5.Erase learning memory (pos='1',neg='-1',both='2'/no='0')?

the system asks then if the user wants to restart learning without considering the positive example (1), negative examples (-1), both (2) or considering the current state of learning (0)

If the user has not **changed of reference sub-graph**, the system updates the sorted a posteriori list of sub-graphs and the relevance feedback measurements (c.f. Section 3). Then, questions 2.1 to 2.5 are iterated and the system keeps learning.

Else, questions 1.5 and 1.6 (from section 1) are iterated, and next, the system asks the following questions:

6.Factor to reduce Sub-Graph list ?

the system asks if the user wants to reduce the size of the a posteriori list considered to remove, according to the currently defined semantic, non-likely sub-graphs

7.Do you want to fix some specific parameters for the search ('y'/'n')?

For experimented user responding 'y' enables to fix the number of branches to consider in the search tree for the similarity model optimization and if the similarity model should use Kullback-Leibler divergence instead of a simple mean calculation However , in almost all cases, it is safer to keep a mean calculation. For novice , please respond 'n'

The system updates the sorted a posteriori list of sub-graphs and the relevance feedback measurements (c.f. Section 3). Then all questions since question 2.1) are iterated.

:**3). System feedback**

After each learning step, the system writes a file on the disk named "./APostList"containing the relevance feedback measurements and the sorted a posteriori collection of sub-graphs. The user can visualize this data, using the GUI (c.f. 4.6.)

b)Input & output formats

- The graph input format is format 5 (c.f appendix A)
- The output file "./APostList" for the result visualization using the GUI is at format 8) (c.f appendix A)

c)Usage recommendations :

Let us point out several factor inducing learning instabilities:

- The present learning system runs in 'real-time', only when using the mean difference instead of the Kullback-Leibler divergence to characterize sub-graph edges (c.f. question 2.7). By the way, bugs may appear when using the latter.
- For an efficient parameter learning, the number of branches of the search tree must be high enough (c.f. question 1.9 and i2.7).Indeed, a low number induces and non-relevant learning (parameter estimation). However, because a too high number of branches induces a very slow calculation, user may balance these effects. (The default number of branches is set to 30)
- The larger the time-window is (c.f. question 1.1) , the easier the learning is. Indeed, when the time window is large, the positive examples are clearly separated from the negative ones, and both semantic are easy to learn.
- It is easy to get a coherent a posteriori list using only the positive semantic (note that in this case a posteriori list = likelihood list).
- When incorporating the negative semantic, the user must be very cautious to give 'very' negative examples before giving examples not very discriminated from the positive ones. Indeed, as the search tree optimization (for parametric similarity model) converges to a local minima, if the system is confused since the beginning, it might estimate wrong parameters for the similarity function and remain in the neighborhood of this false estimate latter on

4.6 GAUSSLUT : GAUSSIAN DYNAMIC ADAPTATION AND QUICKLOOKS GENERATION

4.6.1 Gaussian dynamic adaptation

Program for the histogram dynamic adaptation of image time-series.

This procedure transforms, one after the other, the histograms of the different images (or a spatial window in the images) composing the time-series into a Gaussian distribution. For a given spectral band, each image can either be processed independently from the others or can be processed considering the statistics of all the other images. But as we are interested in an visual homogeneity between the image time samples, we only consider the first case in which the processing is done independently for each image band.

For further details, we refer the reader to the concept description and to section 8.1.2 of [19]

a) Activation syntax

i.The program is in the following directory
../DoxygeneCode/VisualInfMining/HistoDynaAdatp
The name of the compiled program implementing this module is the following :*GaussLUT*

ii.The input/output parameters are designed by the following strings listed on one line, one after the others and separated by a white space

→Required parameters :
- arg[1] input text file name containing on separated lines, the paths of the image bands to convert;
- arg[2] output text file name containing on each line the paths of the images to save on the disk
- arg[3] number of spectral bands x number of images
- arg[4] Number of lines of the images
- arg[5] Number of columns of the images

→Optional parameters for selecting a spatial window in the 'image steak' :
- arg[6] Column offset when reading the image file (default is 0)
- arg[7] Line offset when reading the image file (default is 0)
- arg[8] Number of columns of the image file(default is arg[5])

iii.Activation syntax example

Move to the directory
cd ../DoxygeneCode/VisualInfMining/HistoDynaAdatp

Run the first line of the script *./example.sh* :

./GaussLUT in.txt out.txt 9 100 100 400 200 3000

b) Input & output formats

- The image bands must be in short integers
  (otherwise needs to be recompile changing the PIX variable definition)
- The output normalized image bands are in bytes

4.6.2 Quicklook generation

We use the unix/linux function to convert the steaks of 3 image bands into JPEG RGB images quicklooks

i. Activation syntax example

Move to the directory

cd ../DoxygeneCode/VisualInfMining/HistoDynaAdatp

Run the last lines of the script *./example.sh*:

rawtoppm -bgr 100 100 ../../data/20001015.Graw > ../../data/20001015.ppm
ppmtojpeg ../../data/20001015.ppm > ../../data/20001015.jpg
rm ../../data/20001015.Graw ../../data/20001015.ppm
rawtoppm -bgr 100 100 ../../data/20001025.Graw > ../../data/20001025.ppm
ppmtojpeg ../../data/20001025.ppm > ../../data/20001025.jpg
rm ../../data/20001025.Graw ../../data/20001025.ppm
rawtoppm -bgr 100 100 ../../data/20001031.Graw > ../../data/20001031.ppm
ppmtojpeg ../../data/20001031.ppm > ../../data/20001031.jpg
rm ../../data/20001031.Graw ../../data/20001031.ppm

4.7 GUI : GRAPHICAL USER INTERFACE

This visual interface was developed in order to create a first tool to navigate through the generated image time-series representations, and therefore to validate the results.

This tool enables the visualization and the synchronized navigation through the image time-series representation, the graph of dynamic cluster trajectories and the dynamic classfiles. Furthermore, if the interactive learning C++ program is launch in parallel, the current semantic labeling and the relevance feedback measurements learned by the system can be visualized by this JAVA GUI (The communication between the C and the JAVA program is done using text files).

For further details, we refer the reader to section 8.1.2 of [19].

a) Activation syntax

i.The program is in the following directory
../DoxygeneCode/VisualInfMining/JAVA_GUI1.0

The name of the main compiled class implementing this module is the following *ITS_Graph3.class*

ii.The input/output parameters must be fixed before compilation

- in the *./ITS_Graph3.java* file :
  - at line 76 *private static int dimx=X*;
    where *X* is the number of line of the images
  - at line 78 *private static int dimy=X*;
    where *X* is the number of column of the image;
  - at line 151, *private static String **IMAGEPATH** ="X",*
    where *X* has the following syntax: *directoryPath+imagePrefix*
    and where *directoryPath* is the directory path of the images JPG quick looks of 200x200 pixels, and imagePrefix is the quicklook images name prefix. The image name sufffix must be the image time sample index (e.g. 01,02,...,10,34,etc..)
  - at line 153 priva*te static String **CLASSFILEPATH_MT** = "X",*
    where *X* is the path to the MT classfile of 200x200 pixels
  - at line 155, *private static String **CLASSFILEPATH_TL** = "X",*
    where *X* is the directory path of the classfiles of 200x200 pixels in this directory the classfile must be named with their corresponding dates in format *'yyyymmdd'+'.cl'*, where *'yyyy'* means year,*'mm'* means month,and *'dd'* means day.
    (e.g. 20001015.cl,...,20010216.cl, etc)
  - at line 15*9 String **path**= "X",*
    where *X* is the path to the graph at format 6 (c.f appendix Af)

- in the *./Dates.java* file :
  - fill the ***imageDate*** vector of the class *Dates* with dates of the time samples at the format *'yyyymmdd'*, where *'yyyy'* means year,*'mm'* means month,and *'dd'* means day. (e.g. 20001015,...,20010216, etc).

Note that the ApostList file is read directly in the directory where it was created that is to say : ../SupervisedLearning. For any modification change this directory in the file SBList.java, at line 32.

iii.Activation syntax example:

Once the path fixed in the source, move to the directory

cd ./DoxygeneCode/VisualInfMining/JAVA_GUI1.0

then, compile the source using this command line

javac -classpath ./ ITS_Graph3.java

and run the program *java ITS_Graph3*

iv.Mouse interactions

The frame contains:

- a) the image time-series sub-frame
- b) the graph sub-frame
- c) a TL classfile sub-frame
- d) a MT classfile sub-frame.
- e) a sub-frame for relevance feedbacks visualization
- f) a sub-frame to navigate through the current sub-graph a posteriori collection
- and g) a sub-frame for text values visualization (sub-graph spatio-temporal location, sub-graph posterior probabilities, etc...).

User can interact with almost all these sub-frames. Interactions in the frames :

- -> In the sub-frame a) the user can navigate in time through all the signal representations by pressing buttons. By clicking on a images location, he can naviguate in space and visualize corresponding MT classes, TL classifications, and graph representations
- -> In sub-frame b) the user can visualize the dynamic cluster trajectories associated to the current MT classes (a maximum of 2 MT classes are visualized at the same time in yellow and red). The mutual information is visualized in the same frame. The user can update all the other representations by defining a time coordinate by simple click in the sub-frame. The complexity of the graph can also be fixed interactively, by adjusting the divergence threshold. Note that, consequently the TL classes associated to the MT classes are updated.
- -> In sub-frame c) the user visualize the dynamic classifications. They are constituted of TL classes associated to the 2 different MT classes (common TL classes are displayed in black, and in red and yellow for TL classes associated to only one of the 2 MT classes). Divergence threshold can be tuned to extend or concentrate these dynamic classification.
- -> In sub-frame d) the MT classes are visualized. The user can select the MT classes by clicking on the image interesting locations or by pressing buttons. A maximum of 2 MT classes are visualized in this frame : they are visualized in yellow and red in correspondence with the cluster trajectories and the dynamic classifications.
- -> Sub-frame e) is dedicated to visualize divergence histograms related to the current state of learning. No interaction exist in this frame
- In sub-frame f), to navigate through the current sub-graph a posteriori collection, press buttons are used to read an ./APostlist text file containing relevance feedback measurements and the sub-graphs a posteriori stored collection.
- -> Sub-frame g) is dedicated to the visualization of text parameters. No interaction exist in this frame

b) Input & output formats

- image quiklooks must be in jpg format. Their size must be 200x200 (possibly subsampled)
- TL and MT classfiles must be in byte format. Their size must be 200x200 (possibly subsampled)
- Graph must be at format 6) (c.f appendix A)
- "./APostList" must be at format 8) (c.f appendix A)

5 REFERENCES

1

Centre National d'Etudes Spatiales (feb. 2005), Database for the Data Assimilation for Agro-Modeling (ADAM) project, [Online].
Available : http://medias.obs-mip.fr/adam/web/frameset/f_serveur.html

2

Antunes C.M., Temporal Data Mining: an overview, IST, Lisbon Technical University, Workshop on Temporal Data Mining with the International Conference on Knowledge Discovery and Data Mining, 2001.

3

P. Tan, C. Potter, Finding Spatio-Temporal Patterns in Earth Science Data, department of Computer Science and Engineering, University of Minnesota, Workshop on Temporal Data Mining with the International Conference on Knowledge Discovery and Data Mining, 2001.

4

National Aeronautic and Space Agency - Jet Propulsion laboratory (feb. 2005), Literature database on ocean surface topography from space, [Online]. Available : http://topex-www.jpl.nasa.gov/science/time-series-data.html

5

M. Schroeder, H. Rehrauer, K. Seidel and M. Datcu, Interactive Learning and Probabilistic Retrieval in Remote Sensing Image Archives, IEEE Trans. on Geoscience and Remote Sensing, pp. 2288-2298, 2000.

6

R. Duda, P. Hall and D. Stork, Pattern Classification (2nd edition), Wiley-Interscience, 2000.

7

T.W. Lee, M. Girolami, A.J. Bell, T.J. Sejnowski, A unifying Information-theoretic Framework for Independent Component Analysis, International journal of computers and mathematics with applications, 1999. in press.

8

P.J. Huber, Projection pursuit (with discussion), The Annals of Statistics vol. 13, No.2, pp. 435-475, 1985.

9

C. Posse, An effective Two-dimensional Projection Pursuit Algorithm, Comm. Statist. Simul. Comput., 19, 1143-1164, 1990.

10

J. H. Friedman, Exploratory Projection Pursuit, J. Amer. Statist. Assoc. vol 82, pp. 249-266, 1987.

11

J. Rissanen, Stochastic complexity and modeling, Annals of Statistics, vol. 14, pp. 1080-1100, 1986.

12

H. Bischof, A. Leonardis, and A. Selb, MDL Principle for Robust Vector Quantization, Pattern Analysis and Applications, vol 2, pp. 5972, 1999.

13

C.S. Wallace and D.L. Dowe, Intrinsic Classification by MML - the Snob Program, proc. of the 7th Australian Joint Conf. on Artificial Intelligence, 1994, pp 37-44.

14

P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman, AutoClass: A Bayesian classification system, in Proceedings of the 5th Internatinal Conference on Machine Learning, Ann Arbor, 1988, pp. 54-56.

15

A.P. Dempster, N.M. Laird et D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, B 39, pp 1-38, 1977.

16

S. Kullback, Information theory and statistics, Dover publications Inc., 1968.

17

P. Héas, M. Datcu, Bayesian learning on graphs for reasoning on image time-series, Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Editors: R. Fisher, R. Preuss, U. von Tousaint, American Institute of Physics, AIP Conf. Proc., Vol. 735, pp. 127-137, 2004.

18

P. Héas, M. Datcu, Modeling trajectory of dynamic clusters in image time-series for spatio-temporal reasoning, to be published in IEEE trans. on GRS, 2005.

19

P. Héas, Apprentissage Bayésien de Structures Spatio-temporelles : application a la fouille visuelle d'information dans les séries temporelles d'images satellites, thèse de l'Ecole Nationale de l'Aéronautique et de l'Espace, 2005.

20

H .Bunke and G. Allerman, Inexact graph matching for structural pattern recognition, Pattern Recognition Letters, vol.1, no.4, pp.245-253, 1983.

21

P. Vitanyi, M. Li, Minimum Description Length Induction, Bayesianism, and Kolmogorov Complexity, IEEE Transactions on Information Theory, vol.46, no.2, 446-464, 2000.

22

M Datcu, and K. Seidel, KIM - Knowledge Driven Information Mining in Remote Sensing Image Archives, [Online]. Available : http://earth.esa.int/rtd/IIMCG/documents.html, November 12, 2002.

6 APPENDIX

6.1 A - FORMATS

Format 1 - Image Time Series:

image bands are raw binary files where the pixels are coded either in short integers (2 bytes) or in floats (4 bytes)
Extension : .(tg1,tg2,tg3); (.norm,.var,.evid)

Format 2 - Projection components:

these images are coded in raw binary files where pixels are coded in floats (4 bytes)
Extension : .(PCA1,PCA2,...); (PP1a,PP1b,...)

Format 3 - MT & TL Classfiles :

these images are raw binary files where pixels are coded in unsigned char (1 byte)
Extension : .cl

Format 4 - Attribute files

these objects are associated to the classfiles; each attribute fil is coded in binary floats (4 bytes) using the following syntax:
Extension : .att

The files contains 255 lines of *D*(D+1)* float elements, where *D* is the feature space dimension

- On the 1st line the number of classes followed by zeros
- Each following line correspond to a cluster; it comprises:
  - the cluster mean vector on *D* floats
  - the covariance matrix on *D*D* floats.

Format 5 -Graph file for the supervised learning program:

Let us introduce the following notations : MT for Multitemporal and TL for Time-Localized
This binary object is named *grapheV2*

It is coded on one line, in text format, using the following syntax :

- the number of MT clusters (integer)
- the TL feature space dimension (integer)
- the number of image time samples (integer)
- for each MT cluster *kMT*:
  - the class index (integer)
  - the cluster weight (unsigned integer)
  - for each time sample *t*

- the time sample index (integer)
- the *D* mean vector elements of the projected MT cluster at *t* (floats)
- the *D*D* covariance matrix elements of the projected MT cluster at *t* (floats)
- the number of assigned TL clusters (integer)
- for each assigned TL cluster *kTL_t*
  - -the assigned TL class index(integer)
  - -pixel quantity belonging to classes *kTL_t & kMT(unsigned integer)*
  - -Kullback-Leibler weighted divergence between *kTL-t & kMT (double)*
  - -the *D* mean vector elements of the TL cluster *(float)*
  - -the *D*D* covariance matrix elements of the TL cluster *(float)*
  - -if (t>0)

for each assigned TL cluster at the previous time *kTL_t-1*

- the *kTL_t-1* cluster index (integer)
- the flow of points in that edge (unsigned integer)
- the Kullback-Leibler or the mean vector difference between cluster *kTL_t-1* and *kTL_t* (double)

- if (t>0)

Mutual information between the MT cluster projected at time t-1 and t (float)

Format 6 - Graph file for the Graphical User Interface (GUI):

This binary object is named *grapheV2JAVA*

It is coded on several lines, in text format, using the following syntax :

- the number of MT clusters (integer)
- the TL feature space dimension (integer)
- the number of image time samples (integer)
- for each MT cluster *kMT* :
  - the class index (integer)
  - the cluster weight (unsigned integer)
  - for each time sample *t* :
    - the time sample index (integer)
    - the *D* mean vector elements of the projected MT cluster at *t* (floats)
    - the *D*D* covariance matrix elements of the projected MT cluster at *t* (floats)
    - the number of assigned TL clusters (integer)
    - for each assigned TL cluster *kTL_t*
      - -the assigned TL class index(integer)
      - -pixel quantity belonging to classes *kTL_t & kMT(unsigned integer)*
      - -Kullback-Leibler weighted divergence between *kTL-t & kMT (double)*
      - -the *D* mean vector elements of the TL cluster *(float)*
      - -the *D*D* covariance matrix elements of the TL cluster *(float)*
      - -if (t>0)

for each assigned TL cluster at the previous time *kTL_t-1*

- the *kTL_t-1* cluster index (integer)
- the flow of points in that edge (unsigned integer)
- the Kullback-Leibler or the mean vector difference between cluster *kTL_t-1* and *kTL_t* (double)

-go to a new line

if (t>0) Mutual information between the MT cluster projected at time t-1 and t (float)

Format 7 - Quiklooks for the Graphical User Interface (GUI) :

the quiklooks multispectral images coded in the standard JPG format
Prefixe : date of acquisition at the format «yyyymmdd »
Extension .jpg
Example: 20001015.jpg

Format 8 – A posteriori list for the Graphical User Interface (GUI) :

This file is a text file named *APostList*.
It has the following syntax:
-on 1 line the 7 Kulbback-Leibler divergence measurement (floats)
-on 1 line the mean and the variance of the a posteriori list probabilities (floats)
-on 1 line the 2 convergence measurements (floats)
-on 1 line thenumber of time samples composing the time-window (int)
-for each element of the a posteriori collection of sub graphs on 1 line :

- the index of the associated MT cluster
- the index of the other associated MT cluster (if only one is associated, then this index is the same as the previous)
- the associated a posteriori probability

Format 9 - Delay text file for the Graphical User Interface (GUI) :

This file is a text file named *TimeDelay.txt*
This file stores, on a single line and one after the other, the time differences in days between consecutive images in the time-serie

6.2 B- INSTALLATION

a)Requirements

Requires the g++ compilator and a machine with at least 256 Mo of memory

b)Compilation of the source

i.From the *../Install* directory run the installation script *./InstallPart1.sh*

ii.Then in order to change the format of the input data for the programs of 1) dimension reduction, 2) Gaussian mixture modeling and 3) attribute of classfiles generation. Source modification have to be done. Thus,

1) go in the directory
../DoxygeneCode/Learning/NonSupervisedLearning/DimReduc/
change at line 28 of the file *PCA_PP.cpp* the definition of *PIX* to
#define PIX short int
2) go in the directory
../DoxygeneCode/Learning/ NonSupervisedLearning/GM_Modeling/
change at line 20 of the file *ReferenceVector.hh* the definition of *PIX* to
#define PIX short int
3) go in the directory
../DoxygeneCode/Learning/ NonSupervisedLearning/MakeClusterAtt/
change at line 30 of the file *MakeClusterAtt.cpp* the definition of *PIX* to
#define PIX short int

iii.From the *../Install* directory run the installation script *./InstallPart2.sh*

iv.From the actual *../Install* directory run the installation script *./InstallPart3.sh*