RADIOMETRIC EVOLUTION CLASSIFICATION IN A HIGH RESOLUTION SATELLITE IMAGE TIME SERIES (SITS)

Camille Le Men[1,2], Andreea Julea[3,4,5], Nicolas Méger[3], Mihai Datcu[1,6], Philippe Bolon[3], and Henri Maître[1]

[1]*TELECOM ParisTech, 46 rue Barrault - F-75013 Paris, FRANCE, Email :lemen@telecom-paristech.fr*

[2]*CNES, BPi 1219 18, av. E. Belin, 31401 Toulouse Cedex 9, FRANCE*

[3]*Université de Savoie, Polytech'Savoie/LISTIC - BP 80439 - F-74944 Annecy-le-Vieux Cedex, FRANCE*

[4]*Institutul de Stiinte Spatiale, Bucharest, ROMANIA, Email: andreeamj@venus.nipne.ro*

[5]*Universitatea Politehnica Bucuresti, LAPI, Bucharest, ROMANIA,* [6] *German Aerospace Centrer (DLR), GERMANY*

ABSTRACT

The new generation of Pleiade satellites will be able to acquire High Resolution Satellite Image Time Series (HRSITS). Thus algorithms characterizing time evolutions are required. Since images are not radiometrically calibrated, and since these HRSITS are temporally subsampled, classical time interpolation methods can not be applied. To overcome this problem, we use clustering algorithms and symbol based patterns retrieval algorithms for generating a pixel evolution class image. Those classes allow us to temporally characterize regions obtained by segmentation. Indeed, at these resolutions, relevant information are to be found at the region level. The series we use was acquired for the ADAM (Data Assimilation for Agro-Modelling) project by CNES. Images of the same scene were acquired by SPOT 1,2, and 4, and they are geometrically registered

Key words: satellite image time series, segmentation, K-means, classification, trie, prefix tree .

1. INTRODUCTION

The ADAM Satellite Image Time Series (SITS) consists in thirty eight 20m resolution images of the same zone taken at variable time intervals by three different satellites (SPOT 1, 2 and 4). We used three channels, the wavelength ranges of which are: $[0.5 - 0.59]$, $[0.61, 0.68]$ and $[0.70, 0.59]$ μm. We are interested in the temporal analysis of the series and more precisely in the characteristic evolutions in the series. The variability of the sensors, the atmospheric condition changes, the temporal sub sampling as well as the brutal nature of some observed phenomena (e.g.: harvest) prevent us from using temporal interpolation methods. However, it is visually easy to detect zones evolving similarly. For instance, if one follows two wheat fields through time, even if a a curve does not fit the radiometric evolutions, it seems obvious that they are subject to the same evolution.

In order to overcome the problem of temporal continuity, we consider a purely symbolic framework. In this aim, the image value space is quantized with an Expectation-Maximization (EM) algorithm. In the resulting image series, each pixel has a discrete value corresponding to its cluster number. Then, the evolutions of each pixels are considered: This evolution is a sequence of symbols, each symbol being related to an EM class class. Finally, the pixels are classified according to their evolutions. This task can be a very resource consuming one. Therefore, we propose to use a compact and efficient data structure for storing and classifying evolutions : tries [1, 2], a.k.a. prefix trees. By enriching those structures, we also store pixel positions, which, in turn, allows us to efficiently build images of labels where each label is related to an evolution class. Thus, we obtain a classification image based on pixel evolutions. More details are to be found in Section 2. This technique being pixel based, some zones covered by the classification can present ”pepper and salt” noise which makes the interpretation of the result difficult. Therefore, it is spatially regularized using a segmentation, which, in turn, if independently considered, gives no information about temporal behaviors. The results are easier to interpret as they describe the evolutions of regions. A fusion of the segmentation series and a classification image is done by selecting for each segment, the class the pixels of the regions voted for. It is to note that this method is very easily computable. Details about segmentation series and evolution class generation as well as explanations about the fusion of the segmentation series with evolution classes images are also given in Section 2. Section 3 details experiments on ADAM SITS and gives quantitative and qualitative assessment. Finally, Section 4 concludes this paper and expounds future works.

2. MIXING SEGMENTATIONS AND PIXEL BASED EVOLUTION CLASSIFICATIONS

2.1. Generating segmentation series

In order to describe significant objects of the scene, we need to segment the images. As the dynamic of the images changes through time, segmenting the images independently would not give segmentation of the same scale at different times. Besides, the brutal radiometric changes in time prevent us from using 3D segmentation techniques as well. In order to overcome these problems we use a multi segmentation method presented in [3]. This method extends the classical Minimum Description Length (MDL) image segmentation method to SITS. The MDL segmentation method [4, 5, 6] consists in modelling the image as a partition of connected regions with independent radiometric models. Knowing the partition, the statistics of the regions are easily computable, but there exist many possible partitions of the image. In order to disambiguate this problem, the MDL principle is used. MDL considers segmentation as a coding problem, and tries to minimize the coding length: the image can be coded into a two-part message, the first one being the model part, and the second one, the image knowing its model. This formulation gives a good compromise between the two following extreme cases in which: 1) The partition considered is the absolute over partition (one region per pixel); in this case, the second part of the message is null, but the first part is very large. 2) The partition considered has only one region. In this case, the first part of the message is null, but the second one is very high. Generally, the regions of the ADAM SITS we are interested in are geometrically constant: the rivers generally keep their shapes, as well as the forests, the towns, and generally the fields. But sometimes, the regions can split or merge: for instance during culture growth, crops or cloud occlusion. In order to take these behaviour into account, we consider two hypotheses for two overlapping regions at different times:

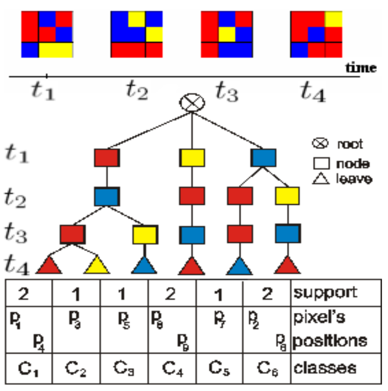
- Either they are independant and the two contours are coded intependantly
- Or they remain constant: the second contour is coded conditionally on the first one. This coding scheme enables some segmentation errors for the regions and some slight changes of the shape of the region.

Region radiometries are considered to be drawn independently through time and are thus coded for all the regions of all images. Within each region, the pixels are considered independent and identically distributed and follow a 3D gaussian model. In order to optimize this description length, a merging algorithm (similar to the one presented in [5]) is used: the algorithm is initialized with series of over segmentations. For each spatially connected regions for all images, the cost (can be negative) of merging as the difference of the description length of the SITS after and before merging is computed. Noticing that only few regions description length change after a fusion enables computing this cost easily. This stack of merges is then sorted according to their merging cost, and the best two candidates are merged. At each step, the cumulative cost (corresponding to the STIS description length minus the initial oversegmentation description length) is computed, and the best series of segmentation is obtained at its minimum.

2.2. Generating an evolution classes image

As written in Section 1, pixel values are, for each image, linked to one single class. More precisely, for each image, a EM algorithm based on a mixture model of the image radiometry is run. Several channels are taken into account and classes number is user-defined.

Then, for each pixel, an ordered list of symbols $s_1, s_2, \ldots, s_I$ is built, where I is the number of images in the SITS and the i^{th} symbol of the list directly refers to the class the pixel belongs to in the i^{th} image of the time series. This list is denoted by $s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_I$ and it is referred to as a *pixel evolution*. If classes, images and pixels are numerous, pixel based evolution classification becomes a very resource consuming task. It either does not fit into memory or it requires, in the worst case scenario, a processing triggering $\frac{P*(P-1)}{2}$ comparisons of pixel evolutions, with P the number of pixels of one image. For more details, the reader is referred to [7]. An



2	1	1	2	1	2	support
p_1 p_4	p_3	p_5	p_6 p_9	p_7	p_2 p_8	pixel's positions
C_1	C_2	C_3	C_4	C_5	C_6	classes

Figure 1. Trie obtained for a toy example (C = 6).

efficient way of processing this type of data is to build a compact data structure such as a trie, a.k.a. prefix tree. This data structure has been proposed by de la Briandais [2] and Fredkin [1]. It has been widely used, for instance for improving telecommunication and data mining techniques (e.g. [8], [9]). For the sake of clarity, we present here tries by adapting the description to our application. A trie can store in an optimized way all different pixels evolutions by storing only once the prefixes that are shared by evolutions. Those evolutions are in turn easy to retrieve as each path from the root to a leaf refers to a given evolution. Thus, if we store into the trie the pixel positions of each evolution class, then we just have to browse once the trie for generating an evolution class image. Let us consider the toy example depicted in Figure 1. In this example, the input dataset is a SITS that contains 4 images of 9 pixels each (p_1 to p_9, in raster order). Three symbols ('red', 'blue' and 'yellow') are defined for describing reflectance values over the whole SITS. These images had been acquired at time t_1, t_2, t_3 and t_4. For each pixel, we insert its pixel evolution into a trie (see Figure 1). If an evolution already exists, i.e. a full path from root node to a leave corresponding to this evolution exists, then no branch or node is created. On the contrary, if there is no corresponding path, then appropriate nodes and branches are generated. As can be observed in Figure 1, a class symbol as well as the instances positions are linked to each leave, . For instance, pixel evolution $red \rightarrow blue \rightarrow red \rightarrow red$ holds for pixels p_1 and p_4 and they fall into class C_1. This corresponds to the left branch of the trie. The first (w.r.t. root node) two nodes of this branch also store the common prefix $red \rightarrow blue$ shared by classes C_1, C_2 and C_3. As can be observed, this data structure efficiently takes part to reduction by storing only once common prefixes. Processing times are good as they are bounded by $P * \frac{1}{2}(s+1)log_s P$, with s the average number of son nodes for a given father node. Implementation details are to be found in [7].

2.3. Fusion of a segmentation series with an evolution class image

The obtained classification is difficult to interpret since it contains "pepper and salt" noise. One would like to be able to deduce from it the radiometric evolutions of the objects of the scene. In order to interpret the classification that way, we propose to use a segmentation and to com-

bine it with the first classification. This problem can be seen as a classifier combination problem. There has been recently a large interest given to this field: [10, 11, 12] among many others.
As mentioned in [10], the classifier combination problem can be divided into three types depending on the information level of the output of the classifiers to combine:

- the abstract level: the classifiers output only the class selected for each sample. Or, for some extension, they output a subset of the possible classes.
- the rank level: each classifier outputs an ordered list (of a subset) of the possible labels for each sample.
- the measurement level: each classifier outputs a confidence measure for each class (of a subset) of the possible labels for each sample.

Our case corresponds to the first type: in each region, each pixel can be considered as a classifier outputting his guess for the class of the region. Selecting a class for the region can thus effectively be seen as a classifier combination problem: each pixel is considered as an elector, voting for the region's evolution among all the possible evolutions.
Let us represent each region R_i $i \in [1, N_r]$ by a $K \times |R_i|$ matrix, $(A_{j,k})$, where $|R_i|$ is the number of pixels in the region, and K the number of evolution classes. Each column vector A_k represents the k^{th} pixel decision: ie it has a unique non zero entry corresponding to its evolution class label. The criteria we consider to affect one evolution class for the region is strictly speaking the "plurality vote": select class k such that $\sum_{j=1}^{|R_i|} A_{j,k} = max_{l\in[1,K]} \sum_{j=1}^{|R_i|} A_{j,l}$, but, it is generally referred to, in the literature, as the majority vote. It can in fact be seen as a majority vote if the number of outcomes is reduced to two: "correct decision" and "wrong decision", thus falling into a binary case in which plurality vote and majority vote are equivalent [13].

3. EXPERIMENTS

3.1. Input data and pre-processing

Before generating the evolution class images, the image value space must be quantized to reduce the size of the tree to build. In that aim, we perform an EM clustering on the images. We chose this clustering method because it is based on a mixture model and enables therefore each data point to be attached to different cluster centers with different weights (the probability of the data point to be attached to one cluster center). This algorithm thus performs well on intricate data. Once the algorithm has converged, we harden the result by assigning to each data point the index of the Gaussian parameters maximizing the a posteriori probability. For further details on EM clustering algorithm, the reader is referred to [14]. In order to choose the number of clusters for each image, we use the MDL principle: for different number of clusters, we compute the EM clustering and the resultant description length. The chosen number of clusters is the one minimizing the description length. The coding of the Gaussian parameters (mean and covariance for each cluster) has the following length:

$$\mathcal{L}(\mathcal{M}) = \frac{K}{2}\left[\frac{d(d+1)}{2} + d\right]\log(N) \tag{1}$$

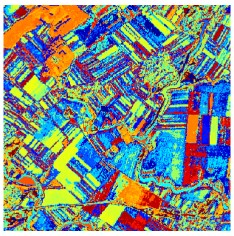


Figure 3. Pixel evolution classification results. The corresponding number of classes is 782823.

where, $n(k)$ is the number of pixels attached to the cluster k, d is the space dimension (3 channels in our case), and N the number of pixels in the images. The fidelity term can be approximated as:

$$\begin{aligned}\mathcal{L}(\mathcal{D}|\mathcal{M}) = & \frac{Nd}{2}(\log(2\Pi)+1) + \\ & \textstyle\sum_{k=1} Kn(k)\left\{\log(|\Gamma_k|) - log(p(\Gamma_k,\mu_k))\right\}\end{aligned} \tag{2}$$

Let us note that the approximation done $(\sum_{x\in c_k} \frac{(x-\mu_k)^T\Gamma_k^{-1}(x-\mu_k)}{2} \approx \frac{n(k)d}{2})$ is justified only in the case where a large number of points are attached to each cluster. For some images, the minimum seems not to be reached although the maximum number of clusters considered is quite high (60). However, there always exist one local minimum. We decided to choose, for each image, the number of clusters corresponding to this first minimum. The chosen number of clusters ranges from 8 to 33. Figure 2a) shows a crop of the 1st, the 11th, and the last original images. The resulting clustering images are shown in Figure2c).

Figure 3 shows the pixel evolution classification, and Figure 2b) shows the segmentation series.

3.2. Results

The final series of object evolution classifications is presented in Figure 2d) for the EM-MDL clustering. The number of evolution classes for all the regions of all the images is 40209 which corresponds to a reduction of 95% of the number of classes before fusion (782823).
In order to quantify the error induced by the fusion, one can compute the increase in the description length of the data knowing the resulting clustering series and the data knowing the input clustering series:

$$\Delta\mathcal{L}(\mathcal{D}|\mathcal{M}) = \sum_{x \text{ changed class}} \log\left(\frac{(x-\mu_b)^T\Gamma_b^{-1}(x-\mu_b)}{(x-\mu_a)^T\Gamma_a^{-1}(x-\mu_a)}\right) \tag{3}$$

where a stands for "after fusion" and b stands for "before fusion". The results are shown in Figure 4. This error is due to the misclassification in the majority vote step. It could probably be reduced a lot by techniques handling better the case where there is more than one representative class. The final object evolution classification can be used to find regions evolving similarly throughout the whole sequence. An example of such regions is given in Figure 5.

4. CONCLUSION

In this paper, we proposed to generate a spatio-temporal description of the SITS by merging an evolution class im-

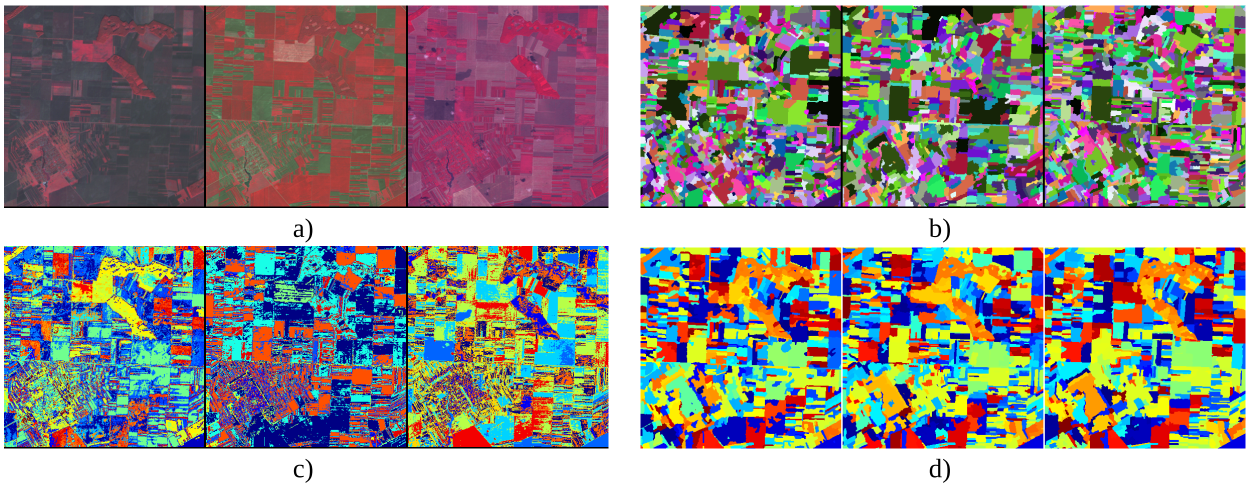


Figure 2. a) 400×400 *crop of the* 1st, 11th*, and last original images of the ADAM Series. (©cnes). b) Input segmentation series. c) EM clustering result. d) Final Result: Object evolution classification for input quantization with EM algorithm.*

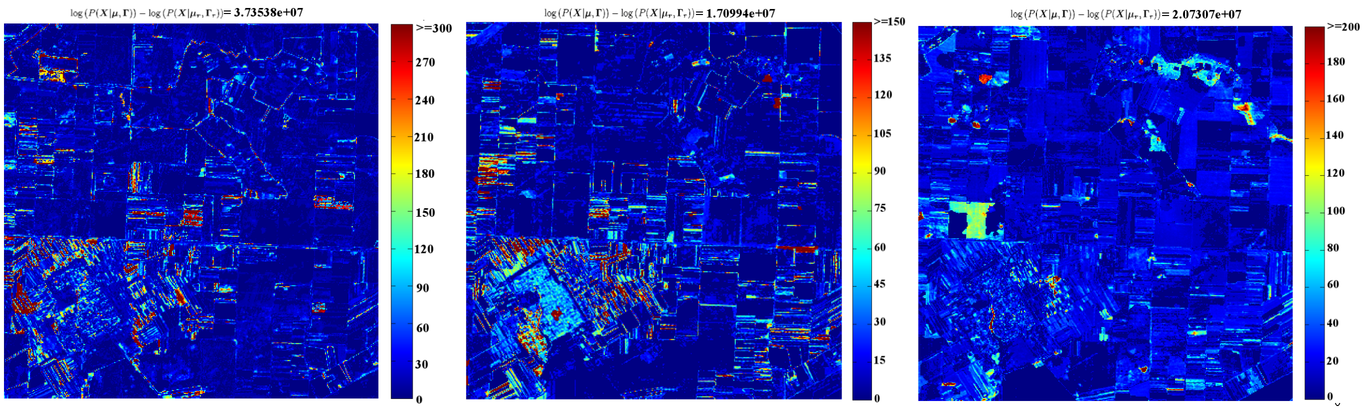


Figure 4. This Figure shows the reconstruction error. $\boldsymbol{\mu} = (\mu_1, ..., \mu_K)$, $\boldsymbol{\Gamma} = (\Gamma_1, ..., \Gamma_K)$*, and the* r *index stands for reconstructed. The first image has far largest reconstruction error.*

age and a segmentation. Final result is simpler than the input classification and adds temporal evolution information to the input segmentation. Furthermore, meaningful objects can be easily identified. For the combination of classifiers, we used a very simple method and however achieved some interesting results. Using a more precise combination framework enabling for instance mixtures of evolution classes to characterize a region would probably give better results. An other possible improvement could be achieved by choosing the best segmentation according to a minimization of the description length of the final result instead of fixing it at the beginning. Finally, considering the regions objects throughout the whole method from the beginning to the end would certainly give more easily interpretable results. In fact using the pixel based evolution classification before fusing it with the segmentation series presents the drawback that there is as much series reconstructions as images. Such a method requires an efficient region matching algorithm.

ACKNOWLEDGEMENTS

The authors thank the ADAM project and the CNES agency for making data available.

REFERENCES

1. E. Fredkin. Trie memory. *Commun. ACM*, 3(9), 1960.
2. R De La Briandais. File searching using variable length keys. In *Proc. of the Western Joint Computer Conference*, pages 295–298, New York, 1959.
3. C. Le Men, H. Maître, and M. Datcu. Minimum description length applied to the spatio-temporal segmentation of high resolution satellite image time series. In *Telecom Technical report*, May 2008.
4. Yvan G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3(1):73–102, 1989.
5. T. Kanungo and al. A fast algorithm for mdl-based multi-band image segmentation. *Proc. IEEE Conf. CVPR*, pages 609–616, 1994.
6. F. Galland and al. Minimum description length synthetic aperture radar image segmentation. *IEEE Transactions on Image Processing*, 12, 2003.
7. A. Julea, N. Méger, and P. Bolon. On mining pixel based evolution classes in satellite image time series. In *Proc. of the 4th Conf. on IIM (ESA-EUSC 2008)*, ESRIN - Frascati, Italy, March 2008.
8. Lih-Chyau Wuu, Tzong-Jye Liu, and Kuo-Ming Chen. A longest prefix first search tree for ip lookup.

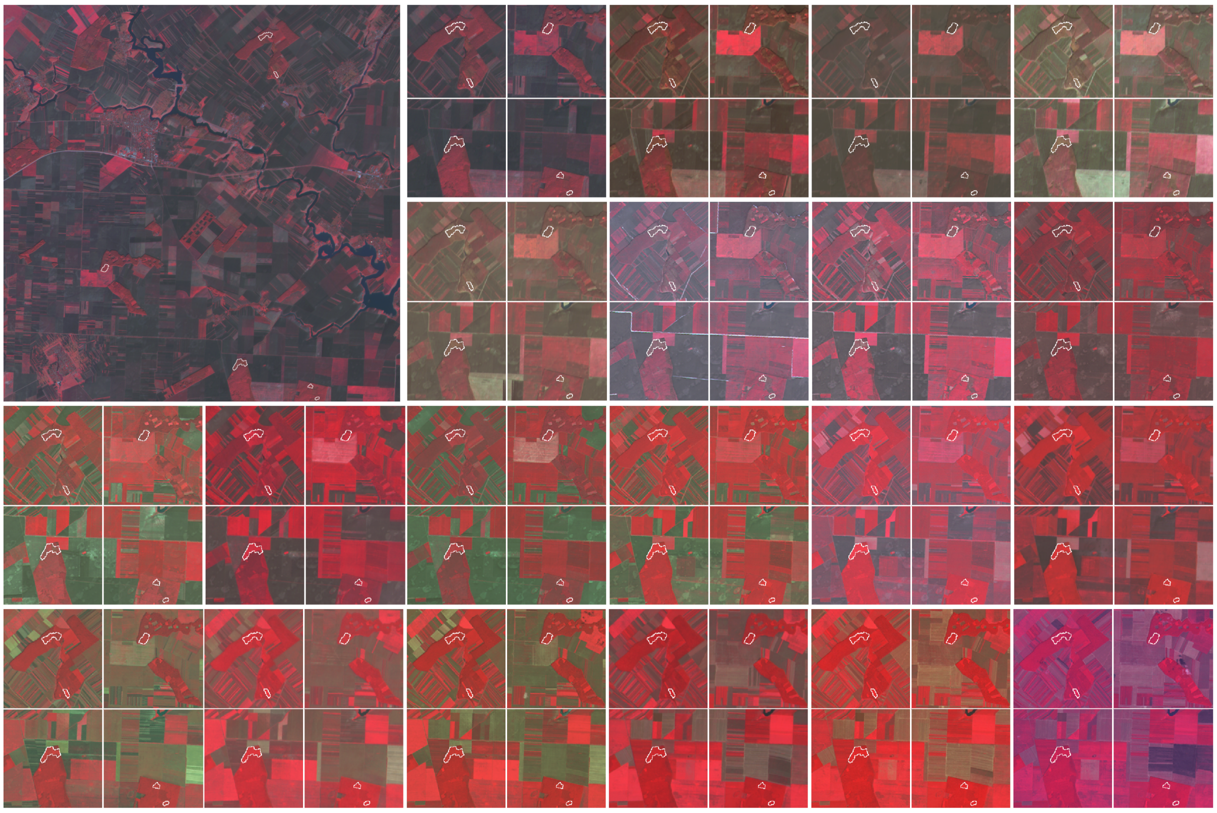


Figure 5. This Figure shows on the entire first original images some regions that have been found to have similar evolution and the corresponding zoomed imagettes for all the images. We can note that although the region seem to have similar behaviour in quasi all the images, in the 11^{th} *image, they effectively behave differenty than all sourounding regions.*

Comput. Networks, 51(12):3354–3367, 2007.

9. J. Pei, B. Han, B. Mortazavi-Asl, and H. Pinto. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proc. of the 17th International Conference on Data Engineering (ICDE'01)*, pages 215–226, 2001.
10. L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. 22(3):418–435, 1992.
11. T.K. Ho. Multiple classifier combination: Lessons and next steps. *Hybrid Methods in Pattern Recognition*, pages 171–198, 2002.
12. Sergey Tulyakov and Venu Govindaraju. Classifier combination types for biometric applications. In *CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, page 58, Washington, DC, USA, 2006. IEEE Computer Society.
13. Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.
14. David J. C. Mackay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, June 2002.