Adaptive Fuzzy Clustering

Nicolas Cebron and Michael R. Berthold
Department of Computer and Information Science, University of Konstanz
78457 Konstanz, Germany
{cebron,berthold}@inf.uni-konstanz.de

***Abstract*— Classifying large datasets without any a-priori information poses a problem especially in the field of bioinformatics. In this work, we explore the task of classifying hundreds of thousands of cell assay images obtained by a high-throughput screening camera. The goal is to label a few selected examples by hand and to automatically label the rest of the images afterwards. Up to now, such images are classified by scripts and classification techniques that are designed to tackle a specific problem. We propose a new adaptive active clustering scheme, based on an initial Fuzzy c-means clustering and Learning Vector Quantization. This scheme can initially cluster large datasets unsupervised and then allows for adjustment of the classification by the user. Motivated by the concept of active learning, the learner tries to query the most 'useful' examples in the learning process and therefore keeps the costs for supervision at a low level. A framework for the classification of cell assay images based on this technique is introduced. We compare our approach to other related techniques in this field based on several datasets.**

I. Introduction

The development of high-throughput imaging instruments, e. g. fluorescence microscope cameras, resulted in them becoming a promising tool to study the effect of agents on different cell types. These devices are able to produce more than 50,000 images per day; up to now, cell images are classified by a biological expert who writes a script to analyze a cell assay. As the appearance of the cells in different assays changes, the scripts must be adapted individually. Finding the relevant features to classify the cell types correctly can be difficult and time-consuming for the user.

The aim of our work is to design a classifier that is both able to learn the differences between cell types and is easy to interpret. As we are dealing with non-computer experts, we need models that can be grasped easily. We use the concept of clustering to reduce the complexity of our image dataset. Cluster analysis techniques have been widely used in the area of image database categorization.

Especially in our case, we have many single cell images with a similar appearance that may nevertheless be categorized in different classes. Another case might be that the decision boundary between "active" and "inactive" is not reflected in the numerical data that is extracted from the cell image. Furthermore, the distribution of the different cell types in the whole image dataset is very likely to be skewed. Therefore, the results of an automatic classification based on an unsupervised clustering may not be satisfactory, thus we need to adapt the clustering so that it reflects the desired classification of the user.

As we are dealing with a large amount of unlabeled data, the user should label only a small subset to train the classifier. Choosing randomly drawn examples from the dataset helps to improve the classification accuracy but needs a large number of iterations to converge. Instead of picking redundant examples, it would be better to pick those that can "help" to train the classifier.

This is why we try to apply the concept of active learning to this task, where our learning algorithm has control over which parts of the input domain it receives information about from the user. This concept is very similar to the human form of learning, whereby problem domains are examined in an active manner.

After introducing the Cell Assay Image Miner in Section II, we propose a new clustering scheme that uses the Fuzzy c-means algorithm with noise detection, which is described in Section III. A sampling scheme that makes use of the fuzzy memberships is proposed in Section IV. We show results in Section V and discuss related work in Section VI, before drawing conclusions in Section VII.

II. Cell Assay Image Mining

In this section we introduce the Cell Assay Image Miner, a software to explore and categorize cell assay images. A typical cell assay image is shown in Figure 1.

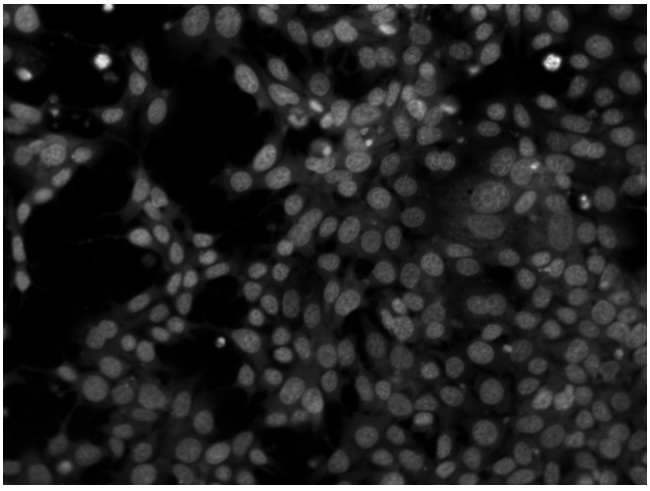

Fig. 1. Original cell image taken by a high-throughput screening microscope camera.

To identify interesting substructures in one image, the original image must be segmented in order to calculate the features

for each cell individually. Unfortunately, the appearance of different cell types can vary dramatically. Therefore, different methods for segmentation have to be applied according to the different cell types. However, the individual cells in one image tend to look similar.

Currently, good results are obtained by an approach that detects a cell nucleus in an image based on a trained neural network. After this step, a region growing is performed in a similar manner to the approach described in [1]. The result of such a segmentation step is shown in Figure 2.

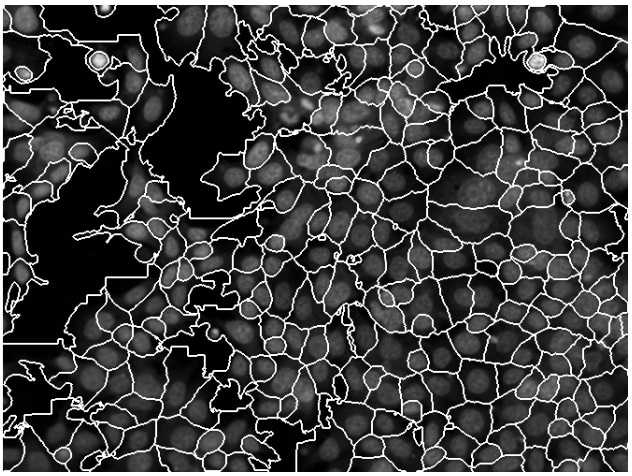

Fig. 2. Segmented cell image.

After the image has been segmented, we can calculate the features on each small subimage of a cell individually. The feature extraction module calculates features of a cell image based on the histogram (first order statistics) or based on the texture (second order statistics). There are also modules for the calculation of Zernike moments [2] and a line feature module that samples points in an image along a vector. The histogram features comprise the mean, variance, skewness, kurtosis, and entropy of the histogram.

The 14 texture features from Haralick [3] represent statistics of the co-occurrence matrix of the gray level image. Four co-occurrence matrices from horizontal, vertical, diagonal, and antidiagonal direction are averaged to achieve rotation invariance. These features provide information about the smoothness, contrast or randomness of the image - or more general statistics about the relative positions of the gray levels within the image.

Currently, the different feature modules are not integrated to form a combined feature vector. One possibility might be to assign weights to each feature in order to control its influence on the classification. At present, we use the feature modules according to requirements of the cell assay images. In Figure 3 we show a table with the single cell images and the Haralick features. The numerical features that we compute based on these images constitute our feature vectors. As we can see from these preprocessing steps, the number of data points may become very large; as we segment thousands of images into small subimages (approximately 200 small cell images per original image), we reach an order of millions of images.

Our goal is to classify the original images by classifying each individual cell within.

At the beginning, we do not have any labeled instances, but we can make use of a biological expert who is able to provide a class label for each cell image that is shown to him. The problem is to classify the whole dataset with as few labeling steps as possible. We have a certain degree of freedom considering the misclassification as the whole image is classified by a majority decision over the small cell images. If a clear majority decision can be made, the image is not considered further. Borderline cases with equal distributions of classes are sorted into a special container to be assessed manually by the biological expert. It becomes apparent that this approach allows for a rather high fault tolerance, as a human will have no objections to labeling a few images by hand rather than risk a misclassification.

In the next section we propose a scheme that tackles this special setting by first clustering the whole unlabeled dataset unsupervised and then assigning class labels to the cluster prototypes. This classification can then be adjusted by the user; we propose a query function that tries to select the most useful examples by taking into account the fuzzy memberships.

III. Fuzzy C-means with Noise Detection

The fuzzy c-means (FCM) algorithm [4] is a well-known unsupervised learning technique that can be used to reveal the underlying structure of the data based on a similarity measure. Fuzzy clustering allows each data point to belong to several clusters, with a degree of membership for each one. We use the extended version from [5] for the added detection of noise.

Let $T = \vec{x}_i\ , i = 1, \ldots, m$ be a set of feature vectors for the data items to be clustered, $W = \vec{w}_k, k = 1, \ldots, c$ a set of c clusters. V is the matrix with coefficients where $v_{i,k}$ denotes the membership of $\vec{x}_i$ to cluster k. Given a distance function d, the fuzzy c-means algorithm with noise detection iteratively minimizes the following objective function with respect to v and w:

$$J_m = \sum_{i=1}^{|T|} \sum_{k=1}^{c} v_{i,k}^m d(\vec{w}_k, \vec{x}_i)^2 + \delta^2 \sum_{i=1}^{|T|} \left(1 - \sum_{k=1}^{c} v_{i,k}\right)^2 \quad (1)$$

$m \in (1, \infty)$ is the fuzzification parameter and indicates how much the clusters are allowed to overlap each other. The first term corresponds to the normal fuzzy c-means objective function, whereas the second term arises from the noise cluster. δ is the distance from every datapoint to the noise cluster c. This distance can either be fixed or can be updated in each iteration according to the average interpoint distances. Objects that are not close to any of the cluster centers $\vec{w}_k$ are therefore detected as having a high membership to the noise cluster. J_m is subject to minimization under the constraint

$$\forall i : 0 \leq \sum_{k=1}^{c-1} v_{i,k} \leq 1 \quad (2)$$

Key	Mask	000|0	000|1	ASM	Contrast	Correla...	Variance	IDFM	SumAv...	SumEnt...	Entropy	DiffEntr...	ICM1	ICM2	Mean
001|001_1				0.003	40.951	8,278,475.228	865.402	0.437	58.379	3,853,049,9...	7.219	3.357	-0.349	0.963	27.476
001|001_2				0.003	43.955	14,312,165....	1,442.186	0.412	74.935	3,850,449,5...	7.199	3.355	-0.395	0.976	35.807
001|001_3				0.006	24.457	3,298,290.137	567.331	0.542	46.609	3,858,187,6...	6.79	3.097	-0.358	0.955	21.262
001|001_4				0.001	95.142	67,850,007....	3,482.665	0.297	115.29	3,828,314,9...	7.798	3.702	-0.424	0.988	54.682
001|001_5				0.001	226.528	298,024,54...	11,173.029	0.197	203.959	3,791,608,8...	7.743	3.739	-0.502	0.996	98.043
001|001_6				0.001	269.863	104,461,02...	4,569.976	0.248	130.187	3,833,704,9...	7.377	3.529	-0.547	0.997	62.369
001|001_7				0.003	43.717	17,088,075....	1,576.672	0.394	78.399	3,854,109,0...	7.046	3.282	-0.439	0.984	36.641
001|001_8				0.003	56.669	20,702,452....	1,854.506	0.315	86.992	3,853,269,5...	6.729	3.192	-0.441	0.983	42.013
001|001_9				0.004	26.218	11,133,549....	1,424.536	0.449	76.741	3,855,797,7...	6.737	3.13	-0.391	0.969	36.068
001|001_10				0.002	34.029	27,050,971....	2,176.543	0.362	91.432	3,846,060,8...	7.428	3.486	-0.404	0.981	42.286
001|001_11				0.005	22.7	5,401,953.392	855.852	0.535	56.229	3,853,994,4...	7.074	3.239	-0.349	0.958	25.966
001|001_12				0.003	75.814	13,149,090....	1,301.413	0.448	64.114	3,842,784,0...	7.483	3.471	-0.403	0.98	30.975

Fig. 3. Table showing each cell with its corresponding mask and numerical features.

FCM is often used when there is no a-priori information available and thus can serve as an overview technique.

IV. From Clustering to Classification

Based on the prototypes obtained from the FCM algorithm, we can classify the dataset by first providing the class label for each cluster prototype and then by assigning the class label of the closest prototype to each datapoint.

Datapoints that are detected as noise are removed because they do not help to enhance the classification. We will give reasons for doing so later.

In order to have enough information about the general class label of the cluster itself that represents our current hypothesis, we perform a technique known as *Cluster Mean selection* [6]. Each cluster is split into subclusters; subsequently, the nearest neighbor of each cluster prototype is selected for the query procedure. If the class distribution within the current cluster is not homogeneous, we replace the prototype with the prototypes of the subclusters. We call this the exploration phase, as we are trying to get an overview of which kind of categories exist in the dataset.

A common problem is that the cluster structure does not necessarily correspond to the distribution of the classes in the dataset. The redefinition of cluster prototypes could increase the classification accuracy. We make use of the Learning Vector Quantization algorithm for this task, which is described in the following section.

A. Learning Vector Quantization

Learning Vector Quantization [7] is a so-called competitive learning method. The detailed steps are given in Algorithm 1. The algorithm works as follows: for each training pattern, the nearest prototype is identified and updated. The update depends on the class label of the prototype and the training pattern. If they possess the same class label, the prototype is moved closer to the pattern, otherwise it is moved away. The learning rate ϵ controls the movement of the prototypes. The learning rate is decreased during the learning phase, a technique known as *simulated annealing* [8]. The LVQ algorithm terminates if the prototypes stop to change significantly. One basic requirement in the LVQ algorithm is

Algorithm 1 LVQ algorithm

1: Choose R initial prototypes for each class $m_1(k), m_2(k), \ldots, m_R(k), k = 1, 2, \ldots, K$, e. g. by sampling R training points at random from each class.
2: Sample a training point $\vec{x}_i$ randomly (with replacement) and let $m_j(k)$ denote the closest prototype to $\vec{x}_i$. Let g_i denote the class label of $\vec{x}_i$ and g_j the class label of the prototype.
3: **if** $g_i = g_j$ **then** {that is they belong to the same class}
4: move the prototype toward the training point: $m_j(k) \leftarrow m_j(k) + \epsilon(\vec{x}_i - m_j(k))$, where ϵ is the learning rate.
5: **end if**
6: **if** $g_i \neq g_j$ **then** {that is they belong to different classes}
7: move the prototype away from the training point: $m_j(k) \leftarrow m_j(k) - \epsilon(\vec{x}_i - m_j(k))$
8: **end if**
9: Repeat step 2, decreasing the learning rate ϵ to zero with each iteration.

that we can provide a class label for each training point $\vec{x}_i$ that is randomly sampled. We assume that the training set is unlabeled - however an expert can provide us with class labels for some selected examples. As we can only label a small set of examples, we need to optimize the queries with a strategy to boost the classification accuracy while keeping the number of queries at a low level. In the next section, we propose a query function that attempts to solve this problem.

B. Selection of Examples based on Fuzziness

The selection of new examples is of particular importance as it influences the performance of the classification. Assuming access to a noiseless oracle it is vital to gain as much as information as possible from the smallest possible number of examples. If we act on the assumption that the underlying structure found by the FCM algorithm already inheres an approximate categorization, we can select further examples by querying data points at the partition boundaries.

We assume that the most informative data points lie between clusters that are not well separated from each other. We call these regions "areas of possible confusion". This coincides with the findings and results in [6] and [9]. Figure 4 demonstrates this setting: There are two clusters; datapoints have been assigned the class label of their closest prototype. As we expect that the distance between similar images in the feature space is small, we can label datapoints close to the prototype with a high confidence, whereas the confidence is lower for points lying between different clusters.

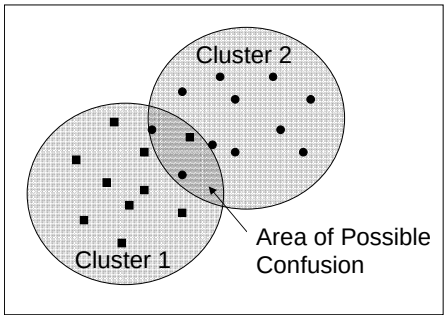

Fig. 4. Two Clusters that overlap

To identify the data points that lie on the frontier between two clusters, we propose a new procedure that is easily applicable in the fuzzy setting. Rather than dynamically choosing one example for the labeling procedure (which would slow down the process), we focus on a selection technique that selects a small batch of N samples to be labeled. Note that a data item $\vec{x}_i$ is considered as belonging to cluster k if $v_{i,k}$ is the highest among its membership values. If we consider the data points between two clusters, they must have an almost equal membership to both of them. The selection is performed in two steps: Initially, all datapoints are ranked according to their memberships to cluster prototypes; subsequently, the most diverse examples are chosen from this pool of examples to avoid choosing points that are too close to each other. The ranking is based on the fuzzy memberships and can be expressed for each datapoint $\vec{x}_i$ as follows:

$$\text{Rank}(\vec{x}_i) = 1 - (\min |v_{i,k} - v_{i,l}|) \quad \forall k, l = 1, \dots c \quad k \neq l \tag{3}$$

Note that we also take into account the class label of each cluster. Only if the clusters correspond to different classes is the rank computed

After all datapoints are ranked, we can select a subset with high ranks to perform the next step: diversity selection. This prevents the active clustering scheme from choosing points that are too close to each other (and therefore are together not that interesting). We refer to the *farthest-first-traversal* [10] usually used in clustering. It selects the most diverse examples by choosing the first point at random and the next points as farthest away from the current set of selected instances. The distance d from a datapoint x to the set S is defined as $d(S, x) = \min_{y \in S} d(x, y)$, known as the *min-max-distance*.

While taking into account samples at the decision boundaries between clusters, the current hypothesis should also be verified. A cluster mean selection step as mentioned in the exploration phase helps to consolidate the classification.

We summarize the procedure we have developed so far in the following section.

C. Adaptive Active Classification

Our adaptive active classification procedure is based on a combination of the techniques that have been mentioned above. All steps are listed in Algorithm 2. We start to cluster our dataset with the fuzzy c-means algorithm, because we expect dense regions in the feature space that are likely to bear the same class label. Therefore, the fuzzy c-means algorithm gives us a good initialization and prevents us from labeling unnecessary instances.

The noise detection in the clustering procedure serves the same purpose: Rare datapoints that represent borderline cases should not be selected, as these noise labels would influence the classification in a negative way. Furthermore, these samples would be useless for the classification. However, note that in this manner, we are able to present unusual and/or outlier cases to the user, that could be interesting to him.

After a batch of N examples has been selected from within each cluster and from the borders of the clusters, the user interaction takes place: the expert has to label each example. The newly labeled samples are then added to the current set of labeled samples L. After this step, the cluster prototypes can be moved based on the training set L.

The question is when to stop the movement of the prototypes. The simulated annealing in the LVQ algorithm will stop the movement after a certain number of iterations. However, an acceptable solution may be found earlier, which is why we propose further stopping criteria:

1) Validity Measures: Can give us information of the quality of the clustering [11]. We employ the within cluster variation and the between cluster variation as an indicator. This descriptor can be useful for the initial selection of attributes.

Algorithm 2 Adaptive Active Clustering Procedure

1: $L \leftarrow 0$
2: Perform the fuzzy c-means algorithm with noise detection (unsupervised).
3: Filter out noise datapoints.
4: **while** Classification accuracy needs improvement **do**
5: Select N training examples within the clusters and from the borders.
6: Ask the user for the labels of these samples, add them to L.
7: Move the prototypes according to L.
8: Decrease the learning rate ϵ.
9: **end while**

Naturally, the significance of this method decreases with the subsequent steps of labeling and adaptation of the cluster prototypes.

2) Classification Gradient: We can make use of the already labeled examples to compare the previous to the newly obtained results. After the labels of the samples inside and between the clusters have been obtained, the cluster prototypes are moved. The new classification of the dataset is derived by assigning to each data point the class of its closest cluster prototype. By comparing the labels given by the user to the newly obtained labels from the classification, we can calculate the ratio of the number of correctly labeled samples to the number of falsely labeled examples.

3) Tracking: Another indicator for acceptable classification accuracy is to track the movement of the cluster prototypes. If they stop moving because new examples do not augment the current classification, we can stop the procedure.

4) Visual Inspection: If the datapoints are linked to images (as in the setting we describe in Section II), we can make use of them. Instead of presenting the numerical features, we select the corresponding image of the data tuple that is closest to the cluster prototype. We display the images with the highest membership to the actual cluster and the samples at the boundary between two clusters if they are in different classes.

V. Experimental Results

As we do not have a completely labeled dataset of cell assay images, we demonstrate the effectiveness of our adaptive clustering scheme first on an artificial dataset and then on the satimage dataset from the UCI Machine Learning Repository [12].

Figure 5 shows the 2-dimensional testdata in a scatterplot. The different gray tones correspond to the different classes in this dataset. This is a typical example for a dataset where the distribution of the classes is skewed. Figure 6 clarifies the difference between random selection on the left side and examples chosen with ranking and diversity selection on the right side. The latter helps the LVQ algorithm to improve the classification accuracy more quickly as can be seen in Figure 7,

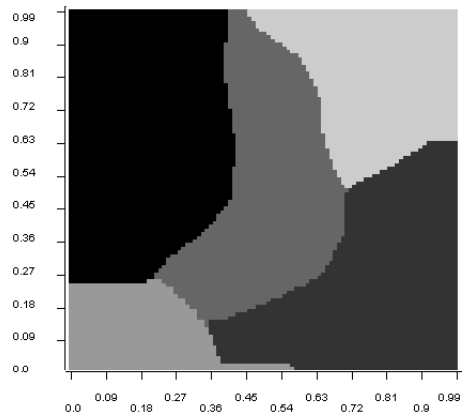

Fig. 5. Scatterplot with 2-dimensional Testdata

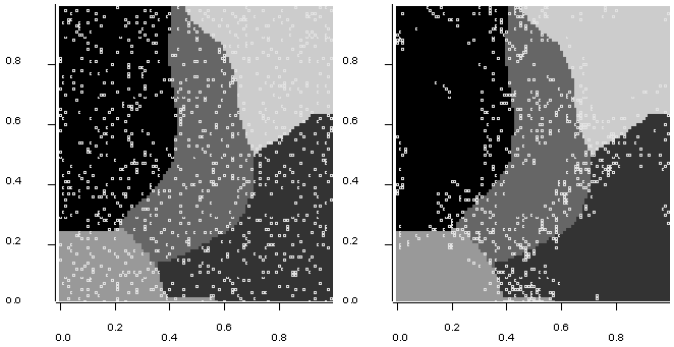

Fig. 6. Different selection techniques: random selection (left) and diversity selection (right)

which shows the classification error in percent over the number of iterations of the LVQ algorithm.

We compared the classification error of the normal LVQ with random selection against our sampling scheme on the satimage dataset that contains 6,435 cases split into 6 classes in a 36-dimensional feature space. Although this dataset does not inherit the structure for which our scheme has been developed, it still performs better than the normal random selection, see Figure 8.

As can be clearly seen, the active selection of datapoints in the learning process of the LVQ algorithm leads to a significantly faster convergence of the classification, especially at the first iterations. This corresponds totally to our objective of keeping user interaction at a low level.

VI. Related Work

There have been a number of approaches to perform partial supervision in the clustering process. In [13], a clustering of the dataset is obtained by first exploring the dataset with a *Farthest-First-Traversal* and providing *must-link* and *cannot-link* constraints. In the second *Consolidate*-phase, the initial neighborhoods are stabilized by picking new examples randomly from the dataset and again by providing constraints for a pair of datapoints.

In [14], an approach for active semi-supervised clustering

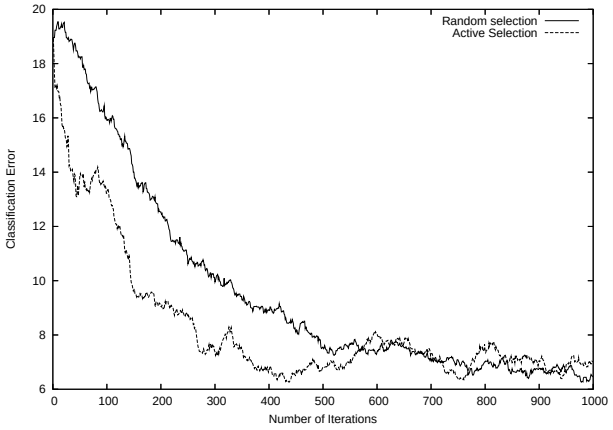

Fig. 7. Active vs. Random Selection

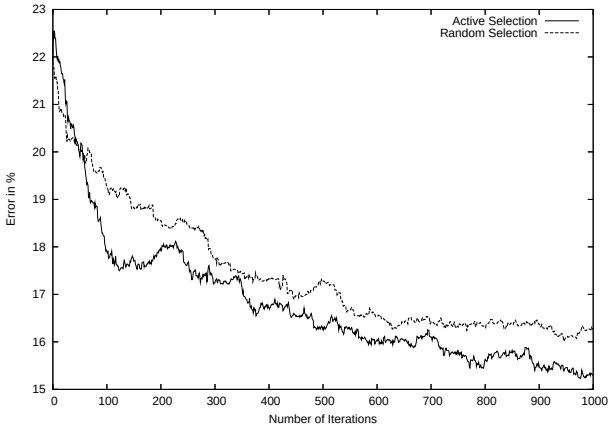

Fig. 8. Active vs. Random Selection on the Satimage Dataset

for image database categorization is investigated. It includes a cost-factor for violating pairwise constraints in the objective function of the Fuzzy c-means algorithm. The active selection of constraints looks for samples at the border of the least well-defined cluster in the current iteration.

In the work of [15], labeled patterns are incorporated in the objective function of the Fuzzy ISODATA algorithm. All these approaches take a set of labeled patterns or constraints as input before the clustering process starts. These samples are selected randomly.

In [16], a very similar approach has been proposed that selects the points to query based on the Voronoi diagram that is induced by the reference vectors. The datapoints to query are selected from the set of Voronoi vertices with different strategies.

The novelty of our approach is in the way that data is preclustered before supervision takes place, which enhances the classification accuracy.

VII. Conclusion

In this work, we have addressed the problem of classifying a large dataset when only a few labeled examples can be provided by the user. We have shown that the fuzzy c-means algorithm is well-suited to be applied to stable initial clustering and that it has the advantage that datapoints on the border can easily be detected by scanning through their memberships to the cluster prototypes. Based on the labels of the selected examples from the borders between clusters and the labeled cluster prototypes, we have proposed to move the cluster prototypes, similar to the Learning Vector Quantization (LVQ) method. We have shown that the misclassification rate can be improved faster than in the normal LVQ method.

Acknowledgment

This work was supported by the DFG Research Training Group GK-1042 ”Explorative Analysis and Visualization of Large Information Spaces”.

References

[1] T. Jones, A. Carpenter, and P. Golland, “Voronoi-based segmentation of cells on image manifolds,” *Computer Vision for Biomedical Image Applications LNCS*, vol. 3765, 2005.
[2] F. Zernike, “Diffraction theory of the cut procedure and its improved form, the phase contrast method,” *Physica*, vol. 1, pp. 689–704, 1934.
[3] R. Haralick, K. Shanmugam, and I. Dinstein, “Textural features for image classification,” *IEEE Transactions on systems, man and cybernetics*, 1973.
[4] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press, 1981.
[5] R. N. Dave, “Characterization and detection of noise in clustering,” *Pattern Recogn. Lett.*, vol. 12, no. 11, pp. 657–664, 1991.
[6] B. Gabrys and L. Petrakieva, “Combining labelled and unlabelled data in the design of pattern classification systems,” *International Journal of Approximate Reasoning*, 2004.
[7] T. Kohonen, *Self-Organizing Maps*. Heidelberg: Springer Verlag, 1995.
[8] S. Kirkpatrick, C. D. G. Jr., and M. P. Vecchi, “Optimization by simulated annealing,” vol. 220, no. 4598, pp. 671–680, 1983.
[9] H. Nguyen and A. Smeulders, “Active learning using pre-clustering,” *ICML*, 2004.
[10] Hochbaum and Shmoys, “A best possible heuristic for the k-center problem,” *Mathematics of Operations Research*, vol. 10, no. 2, pp. 180–184, 1985.
[11] M. Windham, “Cluster validity for fuzzy clustering algorithms,” *Fuzzy Sets and Systems*, vol. 5, pp. 177–185, 1981.
[12] C. B. D.J. Newman, S. Hettich and C. Merz, “UCI repository of machine learning databases,” 1998. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html
[13] S. Basu, A. Banerjee, and R. J. Mooney, “Active semi-supervision for pairwise constrained clustering,” *Proceedings of the SIAM International Conference on Data Mining (SDM-2004)*, 2004.
[14] N. Grira, M. Crucianu, and N. Boujemaa, “Active semi-supervised clustering for image database categorization,” *Content-Based Multimedia Indexing*, 2005.
[15] W. Pedrycz and J. Waletzky, “Fuzzy clustering with partial supervision,” *IEEE Transactions on systems, man and cybernetics Part B: Cybernetics*, vol. 27, pp. 177–185, 1997.
[16] M. Hasenjäger and H. Ritter, “Active learning with local models,” *Neural Processing Letters*, vol. 7, pp. 107–117, 1998.