Clustering with Quantitative User Preferences on Attributes

Adnan El Moussawi *†, Ahmed Cheriat*
*Kalidea Group
Boulogne-Billancourt, France
{aelmoussawi, acheriat}@kalidea.com

Arnaud Giacometti†, Nicolas Labroche†, Arnaud Soulet†
†LI - University of Tours
Blois, France
firstname.lastname@univ-tours.fr

***Abstract*—This paper proposes a new semi-supervised clustering framework to represent and integrate quantitative preferences on attributes. A new metric learning algorithm is derived that achieves a compromise clustering between a data-driven and a user-driven solution and converges with a good complexity. We observe experimentally that the addition of preferences may be essential to achieve a better clustering. We also show that our approach performs better than the state-of-the art algorithms.**

***Keywords*-User preference, clustering, metric learning.**

I. Introduction

Data clustering, one of the most important unsupervised learning problem, is widely used in the field of Customer Relationship Management (CRM). For example, it is commonly used for customer segmentation. Nevertheless, our recent experiments show that many problems remain to be solved. For example, in the context of subspace clustering [1],it has been shown that a large number of interesting clusterings can exists and that it is difficult to automatically select one particular subspace. Moreover, because different users may have different center of interest or preferences, it is important to propose a clustering system that can integrate these subspace preferences when a clustering is built and selected (among all the possible solutions). In that context, the main objective of our work is to show how to take into account the knowledge and preferences of an expert to build a clustering that is a good compromise between a data-driven and a user-driven solution.

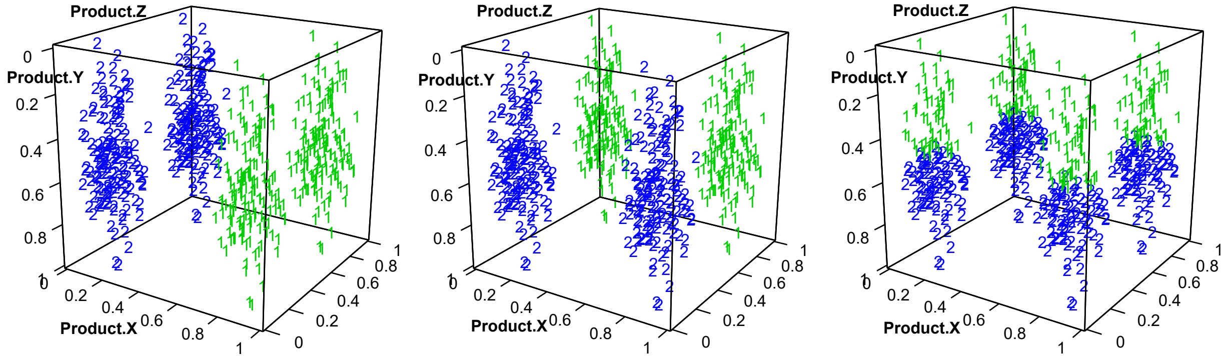
Motivating example. In order to illustrate the objective of our work, we consider the following toy example. Different experts from a marketing agency want to build a customer segmentation based on their purchases of various categories of products named X, Y and Z. As these experts have different professional experiences, we consider that their degrees of interest on categories of product are not the same.

First, consider an expert A that is more interested by purchases of product X, than purchases of products Y and Z. In this paper, we use a quantitative model to represent the preferences of A. More precisely, we assume that each expert assigns to each descriptive attribute a weight proportional to his/her interest for this attribute in clustering analysis. Thus, the preferences of expert A will be represented by the preference vector $W_A = (0.8, 0.1, 0.1)$. If this expert wants to build a customer segmentation with two clusters, using a K-means algorithm with a weighted distance, he/she will obtain the clustering result presented Figure 1a. From another point of view, an expert B with a preference vector $W_B = (0.1, 0.1, 0.8)$ will obtain the clustering presented Figure 1b. It is important to note that the two clusterings obtained by experts A and B are two interesting views of the same data set. Only the preferences of the experts allow to select one of these possible clusterings.

Consider now an expert C with a preference vector $W_C = (0.1, 0.6, 0.3)$. Using a simple K-means with a weighted distance, this expert will obtain the segmentation given by Figure 1c, which is not satisfactory. Indeed, the expert C formulates a high degree of preference on product Y, whereas this attribute does not separate well the set of customers. To avoid this problem, we propose a new approach that take into account the confidence level of an expert in his/her preferences. The confidence level κ is represented by a real value in $[0, 1]$. Thus, if the expert C has a very high confidence in his/her preferences ($\kappa = 1$), he/she will still obtain the segmentation depicted by Figure 1c. However, with a lower confidence level ($\kappa = 0.6$), he/she will obtain the segmentation presented by Figure 1b. Indeed, as the attribute Y does not separate well clusters, our method will tend to favor the other attributes, i.e. the attribute Z whose preference is higher than that of X.

In order to tackle the problems and challenges illustrated by our motivating example, we propose in this paper a new semi-supervised clustering algorithm. More precisely:

- By contrast with previous work that consider constraints specifying if two instances should be in the same cluster or not [2]–[6], we show how to integrate user preferences *on descriptive attributes* in a new clustering objective function. To the best of our knowledge, only the works in [7] addresses the same problem.
- We propose to use a *quantitative* model of preferences to represent the user preferences on attributes. The *qualitative* model used in [7] is more expressive. However, our *quantitative* model is easier to use by an expert.



(a) Clustering with preferences $W_A = (0.8, 0.1, 0.1)$

(b) Clustering with preferences $W_B = (0.1, 0.1, 0.8)$

(c) Clustering with preferences $W_C = (0.1, 0.6, 0.3)$

Figure 1: Customer segmentation using their purchase of products X, Y and Z.

- We present the new clustering algorithm MAPK-means (*Metric Attribute Preferential K-means*) and experiments which show the importance of user preferences on attributes to improve the quality of clustering.

II. Related work

Numerous studies [8], [9] have tackled the problem of feature selection or weighting in the classification, but all these methods are only data-driven. Based on some internal criteria, they select the features that might improve clustering accuracy or interpretability without any user interaction. Subspace clustering [1] also proposes an exploration of different subsets of features where clusters are relevant. As a consequence, there exists a very large number of possible clusterings and the difficulty is to choose from all these potential solutions. Moreover, as with the previous feature weighting methods, these methods are only data-driven and do not rely on user preferences for some feature subset.

Our proposal is part of the family of semi-supervised clustering algorithms that can improve performances and stability from expert knowledge. This knowledge is generally provided as label or pairwise instances constraints that indicate if two objects should be in the same cluster (Must-Link or ML constraints) or not (Cannot Link or CL constraints) and has been adapted to numerous clustering approaches [2]–[6]. Other kinds of constraints have been proposed at the cluster level, mainly to avoid contradictions at the instance level [10], [11] or as relative distance constraints [12] that are more adapted to ranking and instance order preferences. Semi-supervised clustering methods can be categorized in three main families depending if they impose a strict [3] or a soft enforcement of constraints with a penalization term in the objective function [13], or a soft enforcement via the learning of a metric space that minimizes the number of violated constraints [12], [14], [15]. Our approach falls into the the the third category, by taking into account user preferences on attributes. An alternative approach [7], [16] consists in expressing attributes preferences by mean of a triple $(s; t; d)$ which indicates that attribute t is preferred over s with a degree d. Contrary to this approach, our model of quantitative preferences only requires a linear number of preferences, i.e. 1 per attribute. Moreover, in addition to simplifying the interaction with the user, our model leads to a better complexity in metric learning.

III. Problem Statement

Our objective is to propose a new semi-supervised clustering algorithm that can handle quantitative user preferences on attributes. To this aim, we introduce a K-means like algorithm that learns the attribute weights that are the best compromise between the weights provided by the user preferences and the attribute weights that would best fit the natural distribution of data. In the following, a data set $\mathcal{X}$ is a set of N data objects described by M attributes. A partition of K clusters is denoted by $\{\mathcal{X}_j\}_{j=1}^{K}$ and the centroid of cluster $\mathcal{X}_j$ is denoted by c_j.

A. *Quantitative user preferences*

The originality of our approach is to incorporate user preferences on attributes to construct the right partition. We use a *preference vector* $\mathbf{W}^*$ to model preferences where each weight w_i^* represents the weight expressed by the user on the ith attribute. Without loss of generality, we consider that $w_i^* \geq 0$ for all $i \in \{1, \dots, M\}$ such that $\sum_{i=1}^{M} w_i^* = 1$. The set of all preference vectors is denoted by $\mathcal{P}$.

We use the Kullback-Leibler divergence to measure the dissimilarity between two preference vectors. In our case, given the learned vector $\mathbf{W} \in \mathcal{P}$ and a reference vector $\mathbf{P} \in \mathcal{P}$, the dissimilarity between these two vectors is: $D_{KL}(\mathbf{P} \| \mathbf{W}) = \sum_{i=1}^{M} p_i \log\left(\frac{p_i}{w_i}\right)$. In the following, we manipulate two reference vectors $\mathbf{P}$ to express our objective function: the user preferences $\mathbf{W}^*$ and the uniform vector $\mathbf{U} = (1/M, \dots, 1/M)$.

B. *Attribute preferential clustering objective function*

Our clustering objective function consists of three terms that are detailed in the following paragraphs.

Intra-cluster distance: First, as K-means algorithm, we minimize the intra-cluster distance of the clusters $\{\mathcal{X}_j\}_{j=1}^K$. A naive solution could be to directly input the preference vector $\mathbf{W}^*$ to parameter Euclidean distance as follows: $\|x - c_j\|_{\mathbf{W}^*} = \sqrt{\sum_{i=1}^M w_i^*(x[i] - c_j[i])^2}$. However, in this case our solution would only rely on the user expertise and would not take into account the natural distribution of the data. As a side effect, we could output a poor clustering if the user preference vector does not sufficiently discriminate between the data objects (see Figure 1c as a typical example). Consequently, we propose to learn a vector $\mathbf{W} \in \mathcal{P}$ that performs a projection of the initial data space so that the clusters are more compact and well separated in the new space. Thus, we want to minimize: $\sum_{j=1}^K \sum_{x\in\mathcal{X}_j} \|x - c_j\|_{\mathbf{W}}^2$.

Deviation from attribute preferences: Second, we want that the learned vector $\mathbf{W}$ deviates as less as possible from $\mathbf{W}^*$ in order to respect user preferences. Thus, it is necessary to introduce a penalty term to reduce the dissimilarity of $\mathbf{W}$ with $\mathbf{W}^*$: $D_{KL}(\mathbf{W}^*\|\mathbf{W})$.

Regularization term: Third, we add a regularization term that prevents the vector $\mathbf{W}$ to deviate too much from a traditional K-means where all attributes have equal weights. This idea can be formulated as the divergence between the vector to learn and a uniform vector $\mathbf{U} = (1/M, \dots, 1/M)$: $D_{KL}(\mathbf{U}\|\mathbf{W})$.

By combining these three terms, it is possible to define an attribute preferential clustering objective function that expresses a compromise:

$$\begin{aligned}\mathcal{I}_{map} =\ & \alpha\Big(\mathcal{Z}\sum_{j=1}^K\sum_{x\in\mathcal{X}_j}\|x - c_j\|_{\mathbf{W}}^2\Big) \qquad (1)\\ & + (1-\alpha)\Big(\ \kappa D_{KL}(\mathbf{W}^*\|\mathbf{W}) +\ (1-\kappa)D_{KL}(\mathbf{U}\|\mathbf{W})\Big)\end{aligned}$$

where $\mathcal{Z} > 0$, $\alpha \in [0,1]$ and $\kappa \in [0,1]$. Note that $\mathcal{Z}$ is a normalizing constant between intra-cluster distance and other terms because the parameterized Euclidean distance and the Kullback-Leibler divergence have really different ranges (see Algorithm 1 that discusses how to set $\mathcal{Z}$). The objective function of equation 1 depends on two important parameters:

- **Intra-cluster distance weight $\boldsymbol{\alpha}$:** This parameter controls the importance of data compared to that of user preferences. In practice, α is set to an appropriate default value (see Section V).
- **Confidence level $\boldsymbol{\kappa}$:** the user-specified parameter κ gives the importance of his/her preferences. When $\kappa = 1$, the regularization term is not used. The user forces the method to meet his/her preferences. When $\kappa = 0$, user preferences are ignored.

Given a set of data points $\boldsymbol{\mathcal{X}}$, a number of clusters $\boldsymbol{K \geq 1}$, a preference vector $\mathbf{W}^* \in \boldsymbol{\mathcal{P}}$, $\boldsymbol{\alpha \in [0,1]}$ and $\boldsymbol{\kappa \in [0,1]}$, find a $\boldsymbol{K}$-partition $\boldsymbol{\{\mathcal{X}_j\}_{j=1}^K}$ of data minimizing the objective function $\boldsymbol{\mathcal{I}_{map}}$ while learning a vector $\mathbf{W} \in \boldsymbol{\mathcal{P}}$.

IV. MAPK-means Algorithm

A. Reformulation with a Lagrange multiplier

As mentioned in Section III-A, all preference vectors of $\mathcal{P}$ are such that each weight is positive and the sum of weights equals to 1. In particular, the learned vector $\mathbf{W}$ in objective function $\mathcal{I}_{map}$ has to satisfy these constraints:

$$\min_{\mathbf{W}} \mathcal{I}_{map} \qquad \text{subjectto} \sum_{i=1}^M w_i - 1 = 0;\ w_i > 0; \\ \text{for all } i \in \{1,\dots,M\} \qquad (2)$$

We introduce a Lagrange multiplier λ and consider the following function: $\mathcal{I}'_{map} = \mathcal{I}_{map} + \lambda\Big(\sum_{i=1}^M w_i - 1\Big)$. If $\mathbf{W}$ minimizes $\mathcal{I}_{map}$, then there exists a value of λ such that $\mathbf{W}$ is a stationary point for $\mathcal{I}'_{map}$. The stationary point is the point where the partial derivatives of $\mathcal{I}'_{map}$ is zero:

$$\begin{aligned}\frac{\partial\mathcal{I}'_{map}}{\partial w_i} =\ & \alpha\mathcal{Z}\sum_{j=1}^K\sum_{x\in\mathcal{X}_j}\|x[i] - c_j[i]\|^2\\ & - (1-\alpha)\big(\kappa\frac{w_i^*}{w_i} + (1-\kappa)\frac{1}{Mw_i}\big) + \lambda = 0\end{aligned}$$

Assuming that $S_i = \sum_{j=1}^K \sum_{x\in\mathcal{X}_j}\|x[i] - c_j[i]\|^2$ is the total intra-cluster distance on the i-th attribute. We rewrite the above equation for obtaining the update of weight w_i:

$$w_i = \frac{(1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)}{\alpha\mathcal{Z}S_i + \lambda} \qquad (3)$$

The update of weight w_i is central for learning the metric as depicted by the next section. It is easy to see that the lower the variance S_i, the higher the weight of the attribute w_i. When κ is set to 1, only the preferences are used. Conversely, when κ is zero, user preferences are not considered.

B. Algorithm derivation

Our algorithm follows the scheme introduced in [14] consisting in 3 phases: 1) points assignment, 2) centroid re-estimation and 3) metric learning. More specifically, for a given data set $\mathcal{X}$, a number of clusters $K \geq 1$, a preference vector $\mathbf{W}^* \in \mathcal{P}$, a confidence level $\kappa \in [0,1]$ and an intra-cluster distance weight $\alpha \in [0,1]$, the algorithm MAPK-means (*Metric Attribute Preferential K-means*, provided by Algorithm 1) returns a K-partition $\{\mathcal{X}_j\}_{j=1}^K$ minimizing the objective function $\mathcal{I}_{map}$ by learning a vector $\mathbf{W}$.

Algorithm initialization: We use the same initialization as K-means++ [17] (line 1). The weights of attributes for $\mathbf{W}$ are initially equally distributed (line 2): $w_i = \frac{1}{M}$ for $i \in \{1,\dots,M\}$. Finally, $\mathcal{Z}$ is initialized such that the intra-cluster distance and the other terms have a similar impact during the weight update of w_i (see 3) when $\alpha = 0.5$. For this, we choose a $\mathcal{Z}$ value as our update that is identical to that of MPCK-means [14] when $\alpha = 0.5$ (line 3).

Algorithm 1 MAPK-means

```
input a data set $\mathcal{X}$, a number of clusters $K$,
    a preference vector $\mathbf{W}^*$, $\kappa$, $\alpha$
output a partition $\{\mathcal{X}_j\}_{j=1}^K$ and a learned vector $\mathbf{W}$
1: Get $K$ center $\{c_j\}_{j=1}^K$ with K-means++
2: Initialize $\mathbf{W} := (1/M, \dots, 1/M)$
3: Initialize $\mathcal{Z} := \sum_{i=1}^M \frac{\kappa w_i^* + (1-\kappa)/M}{S_i}$
4: repeat
5:     // update the partition $\{\mathcal{X}_j\}_{j=1}^K$
6:     $\mathcal{X}_j := \{x \in \mathcal{X} : \arg\min_{l \in \{1,\dots,K\}} \|x - c_l\|_{\mathbf{W}}^2 = j\}$ for $j \in \{1, \dots, K\}$
7:     $c_j[i] := \frac{\sum_{x \in \mathcal{X}_j} x[i]}{|\mathcal{X}_j|}$ for $i \in \{1, \dots, M\}$ and $j \in \{1, \dots, K\}$
8:     // update the vector $\mathbf{W}$
9:     Compute $\lambda$ using a dichotomic search
10:    $w_i := \frac{(1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)}{\alpha \mathcal{Z} S_i + \lambda}$ for $i \in \{1, \dots, M\}$
11: until $\{\mathcal{X}_j\}_{j=1}^K$ remains unchanged
12: return $\{\mathcal{X}_j\}_{j=1}^K$ and $\mathbf{W}$
```

Cluster assignment: The assignment step is the same as K-means (line 5-6), with the only difference that the distances between points and centroid are parameterized with a vector $\mathbf{W}$. Each point is assigned to the nearest cluster (line 6). This assignment reduces the intra-cluster distance and it also minimizes the objective function $\mathcal{I}_{map}$.

Centroid re-estimation: We update the center of each cluster by calculating the centroid for each attribute i (line 7). Unlike some approaches, the calculation of the centers is insensible to the cluster assignment step.

Metric learning: In this step, MAPK-means learns the right metric by updating the vector $\mathbf{W}$ that minimizes the objective function $\mathcal{I}_{map}$ (line 8-10). As explained in Section IV-A, the update of $\mathbf{W}$ is obtained by taking the derivative $\frac{\partial \mathcal{I}_{map}}{\partial w_i}$ equal to 0. In order to get the exact update of $\mathbf{W}$, we have to compute the Lagrange multiplier λ (see 3). We introduce $p_i = (1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)$ as numerator part and $q_i = \alpha \mathcal{Z} S_i$ as denominator part (excluding λ). Then, equation 3 becomes: $w_i = \frac{p_i}{q_i + \lambda}$ and the calculation of the λ consists in solving the following equation: $\sum_{i=1}^M \frac{p_i}{q_i + \lambda} = 1$.

We use a dichotomic search to determine an approximate solution to this equation (line 9). Consequently, it is necessary to bound λ to initialize this search:

Property 1. *The Lagrange multiplier λ is bounded as follows:*

$$\underbrace{-\min_i(q_i)}_{\inf_\lambda} \leq \lambda \leq \underbrace{\sum_{i=1}^{M} \min_i(p_i) - \max_i(q_i)}_{\sup_\lambda}$$

As the three steps of MAPK-means decrease $\mathcal{I}_{map}$ (which is bounded by 0), MAPK-means converges to a locally optimal solution in a finite number of steps. Besides, its time complexity is $O(i(NKM + NM + jM))$ where i is the number of iterations and j the number of dichotomic search iterations. This is less than the complexity of [7] where the computation of weights optimization is quadratic with $P + M$ (where P is the number of preferences which is upper bounded by M^2).

V. Experiments

In this section, we compare our new MAPK-means to the method introduced in [7] and show that we achieve slightly better results, but solved more efficiently and depending on a single parameter κ that can be easily set and understood by a user. We performed experiments on multivariate attributes data sets from UCI repository[1] for the ease of reproducibility and comparison with other approaches like [7]. To evaluate our experiments, we use the Normalized Mutual Information (NMI) [7] which is a quality index that measures the agreement between two partitions. Its value ranges from 0 to 1: 0 indicates that the two partitions are completely independent and 1 means that they are identical.

Experimental setting: For the purpose of the experiment, we replicate the same protocol as [7]. We first define the *natural* most interesting attributes by computing a weight vector $\tilde{W}$ using the inverse *intra-cluster distortion* Γ_i computed for each attribute. More precisely, the weight of each attribute is defined as follows: $\tilde{w}_i = \frac{\Gamma_i}{\sum_{d=1}^M \Gamma_d}$. In our approach, this weight vector $\tilde{W}$ is used to initialize our preference vector $\mathbf{W}^*$, i.e. $\mathbf{W}^* = \tilde{W}$. Then, similarly to [7], in order to select the best clustering over different runs and different values of $\kappa \in [0, 1]$, we consider the one that minimizes the value of our objective function (see Equation 1). Finally, we set $\alpha = 0.5$ and run 100 tests to ensure the significance of the results. Preliminary experimental study (not presented here due to lack of space) has shown that for this value of α, it is always possible to achieve good clustering results and that this choice is appropriate.

Results: We compare the performances of our MAPK-means algorithm with CFP algorithm introduced in [7]. Compared with our approach, CFP uses a *qualitative* model of preferences on attributes rather than a *quantitative* model using weights on attributes. However, similarly to our proposal, [7] learns a metric parameterized by an attribute feature weight vector, i.e. the most appropriate weight vector with respect to the data set and the user preferences.

The clustering results on all the data sets are shown in Table I. This table compares the clustering results in terms of NMI of the algorithms K-means, K-means with a weighted distance, CFP, for which we present only the best result obtained in [7] (using different values of their parameters m) and finally MAPK-means for which we provide several results, obtained respectively when:

[1] archive.ics.uci.edu/ml/datasets.html

Table I: NMI values for clustering results on K-means, K-means with a weighted distance, CFP [7] and our algorithm MAPK-means. The results of MAPK-means are obtained with κ =0, κ =1 and $\kappa \in [0, 1]$ which maximizes the NMI.

	K-means	*WK-means*	*CFP*	*MAPK-means*			
				$\kappa = 0$	$\kappa = 1$	$\kappa \in [0, 1]$	
Iris	0.742	0.758	**0.864**	0.778	**0.864**	**0.864**	1
Optdigits	**0.756**	0.743	0.715	0.655	**0.720**	**0.720**	1
Pendigits	0.682	0.710	0.707	0.698	**0.718**	**0.735**	0.95
Pgblocks	0.150	0.149	**0.204**	0.107	0.202	**0.204**	0.68
Vowel	0.415	0.397	0.424	0.387	**0.453**	**0.473**	0.83
Wdbc	0.623	0.613	0.628	0.605	**0.665**	**0.677**	0.78
	NMI					$\max(NMI)$	κ

- $\kappa = 0$: this result is equivalent to the result obtained using MPCK-means [14] with metric learning but without ML and CL constraints.
- $\kappa = 1$: the NMI value is obtained when we enforce the user preferences.
- $\kappa \in [0, 1]$: we show the value of NMI of the clustering that maximizes the objective function (see Equation 1). We also give the associated value of parameter κ.

As can be seen in Table I, the best NMI values obtained with CFP and MAPK-means are very similar on *Iris* and *Pgblocks* data set. With the *Optdigits* data set, K-means gives the best NMI value; however, our algorithm MAPK-means outperforms CFP. For all other data sets, the quality of the clusters produced by MAPK-means is better than the quality of clusters produced by CFP, MPCK-means without instance constraints (i.e. MAPK-means with $\kappa = 0$) and a basic K-means. These results also show that even a K-means whose metric is set with the *relevant* weights cannot compete with MAPK-means. Finally, these experiments show that the best clustering quality can be achieved with a weight κ in [0,1], that is to say using at the same time the user preferences and a regularization term.

VI. Conclusion

We propose a metric learning based clustering method that allows the user to express quantitative preferences on attributes. User preferences are formulated as a simple vector which is taken into account by the objective function. We demonstrate that this quantitative model of preferences leads to an efficient metric learning step iterated by our algorithm MAPK-means. Furthermore, experimental results illustrate the positive impact of user preferences on clustering quality and on helping the method finding the right subspace. We also observe that the best clustering result is not achieved by the fully data-driven approach, nor with the fully user-driven one. Finally we show that MAPK-means generally performs better than other algorithms of the literature.

Acknowledgment

We would like to thank Kalidea Group and the ANRT[2] for financial support as part of the thesis (2014/0658).

[2] http://www.anrt.asso.fr/

References

[1] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 90–105, Jun. 2004.

[2] I. I. Davidson and S. Basu, "A survey of clustering with instance level constraints," *ACM Transactions on Knowledge Discovery from data*, pp. 1–41, 2007.

[3] S. Basu, A. Banerjee, and R. J. Mooney., "Semi-supervised clustering by seeding," in *Proc. of the 19th ICML*, 2002.

[4] T. F. C. oes, E. R. Hruschka, and J. Ghosh, "A study of k-means-based algorithms for constrained clustering," *Intelligent Data Analysis*, vol. 17, no. 3, pp. 485–505, 2013.

[5] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *Proc. of KDD*, 2010, pp. 563–572.

[6] C. Ruiz, M. Spiliopoulou, and E. Menasalvas, "Density-based semi-supervised clustering," *Data Mining and Knowledge Discovery*, vol. 21, no. 3, pp. 345–370, 2010.

[7] J. Sun, W. Zhao, J. Xue, Z. Shen, and Y. Shen, "Clustering with feature order preferences," *Intelligent Data Analysis*, vol. 14, pp. 479–495, 2010.

[8] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering: Algorithms and Applications*. Chapman and Hall/CRC, 2013, pp. 29–60.

[9] V. Kumar and S. Minz, "Feature selection: A literature review," *Smart CR*, vol. 4, no. 3, pp. 211–229, 2014.

[10] A. Dubey, I. Bhattacharya, and S. Godbole, "A cluster-level semi-supervision model for interactive clustering," in *ECML PKDD*, Berlin, Heidelberg, 2010, pp. 409–424.

[11] H. Liu and Y. Fu, "Clustering with partition level side information," in *IEEE ICDM*, Nov 2015, pp. 877–882.

[12] E. Y. Liu, Z. Guo, X. Zhang, V. Jojic, and W. Wang, "Metric learning from relative comparisons by minimizing squared residual," in *Proc. IEEE 12th ICDM*, 2012, pp. 978–983.

[13] A. Bouchachia and W. Pedrycz, "A semi-supervised clustering algorithm for data exploration," in *Proc. of the 10th IFSA*, 2003, pp. 328–337.

[14] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *In Proc. of the 21st ICML*. ACM, 2004, p. 11.

[15] D. Klein, S. Kamvar, and C. Manning, "From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering," in *Proc. of the 19th ICML*, 2002, pp. 307–314.

[16] J. Wang, S. Wu, and G. Li, "Clustering with instance and attribute level side information," *Int. Journal of Computational Intelligence Systems*, vol. 3, no. 6, pp. 770—785, 2010.

[17] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. Symp. Discrete Algorithms*, 2007.