Clustering with Repulsive Prototypes

Roland Winkler, Frank Rehm, and Rudolf Kruse

Abstract Although there is no exact definition for the term *cluster*, in the 2D case, it is fairly easy for human beings to decide which objects belong together. For machines on the other hand, it is hard to determine which objects form a cluster. Depending on the problem, the success of a clustering algorithm depends on the idea of their creators about what a cluster should be. Likewise, each clustering algorithm comprises a characteristic idea of the term cluster. For example the fuzzy c-means algorithm (Kruse et al., Advances in Fuzzy Clustering and Its Applications, Wiley, New York, 2007, pp. 3–30; Höppner et al., Fuzzy Clustering, Wiley, Chichester, 1999) tends to find spherical clusters with equal numbers of objects. Noise clustering (Rehm et al., Soft Computing – A Fusion of Foundations, Methodologies and Applications 11(5):489–494) focuses on finding spherical clusters of user-defined diameter.

In this paper, we present an extension to noise clustering that tries to maximize the distances between prototypes. For that purpose, the prototypes behave like repulsive magnets that have an inertia depending on their sum of membership values. Using this repulsive extension, it is possible to prevent that groups of objects are divided into more than one cluster. Due to the repulsion and inertia, we show that it is possible to determine the number and approximate position of clusters in a data set.

Keywords Air traffic management · Fuzzy c-Means · Noise clustering · Repulsive prototypes.

1 Introduction

Prototype-based clustering algorithms have one thing in common: they require knowledge about the expected number of data clusters in advance. Even if this information is at hand, initialization of the prototypes has still a strong influence on the

Roland Winkler (✉)
German Aerospace Center, Berlin, Germany, e-mail: roland.winkler@gmail.com

A. Fink et al., (eds.), *Advances in Data Analysis, Data Handling and Business Intelligence*, Studies in Classification, Data Analysis, and Knowledge Organization,
DOI 10.1007/978-3-642-01044-6_19,

quality of the clustering result. So far, the problem of finding the correct number of clusters and a good initialization for the prototypes cannot be solved analytically, hence, experts or heuristics are needed.

In this paper, we present a method that makes use of available information, such as the expected size and separation of clusters in a data set, to gain knowledge about the number of clusters and their approximate position. This is done by using repulsive prototypes. The repulsive force prevents that prototypes come close to each other which leads to well separated prototypes. The result can be used to initialize (non-repulsive) clustering algorithms.

This paper is structured in four parts. The next part contains a brief description of fuzzy c-means and noise clustering, which will be used to introduce repulsive prototypes later (Bezdek, 1981; Dave & Krishnapuram, 1997). In Sect. 3, we will introduce the mathematical background and the usage of repulsive prototypes. In Sect. 4 we will present results on a practical application. Finally, we conclude with Sect. 5.

2 Fuzzy c-Means and Noise Clustering

Repulsive prototypes extend the concept of fuzzy c-means (FCM) and derivatives, e.g., noise clustering (NC). Although both algorithms are very well known, some mathematical details are needed in the next section, which makes it necessary to repeat them at this point. Let $\boldsymbol{X} \subset V$ be a finite set of data objects of a vector space V with $|X| = n$. The clusters are represented by a set of prototypes $\boldsymbol{B} = \{\beta_1, \ldots, \beta_m\} \subset V$ which can be initialized randomly. Only the number of prototypes m must be known in advance. Let $1 < \omega \in \mathbb{R}$ be the fuzzifier and $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ be the partition matrix with $u_{ij} \in [0, 1]$ and $\forall j : \sum_{i=1}^{m} u_{ij} = 1$. And finally, let $d : V \times V \longrightarrow \mathbb{R}$ be a distance function with its abbreviation $d_{ij} = d(\beta_i, x_j)$.

Fuzzy c-means is defined as an objective function $\boldsymbol{J}$ that is to be minimized

$$\boldsymbol{J}(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{B}) = \sum_{i=1}^{m} \sum_{j=1}^{n} u_{ij}^{\omega} d_{ij}^2.$$

The minimization of J is done by iteratively updating the members of U and B and is computed using a Lagrange extension to hold the side constraint of $\sum_{i=1}^{m} u_{ij} = 1$. The iteration steps are denoted by a time variable $t \in \mathbb{N}$ denoting $t = 0$ as the initialization step:

$$u_{ij}^{t+1} = \frac{(d_{ij}^t)^{\frac{2}{1-\omega}}}{\sum_{k=1}^{m} \left((d_{kj}^t)^{\frac{2}{1-\omega}} \right)}, \tag{1}$$

$$\beta_i^{t+1} = \frac{\sum_{j=1}^{n} \left(u_{ij}^t\right)^{\omega} x_j}{\sum_{j=1}^{n} \left(u_{ij}^t\right)^{\omega}}. \tag{2}$$

For noise clustering, an additional cluster is specified which is represented by a virtual prototype β_0 which has no location in V. Instead, it has a constant $0 < \nu \in \mathbb{R}$ distance to all data objects: $\forall j : d_{i0} = d(\beta_0, x_j) = \nu$ which is called noise distance. β_0 is not represented as a member of V, and so, it is not updated during the iteration process. The noise prototype is introduced to assign higher membership degrees to the noise cluster for all data objects whose distance to regular prototypes exceeds the noise distance. This favors regular prototypes to be better placed in the center of data clusters without being heavily attracted by noise data.

3 Repulsive Prototypes

Unfortunately, noise clustering as described above, has certain disadvantages when it comes to separation of clusters. It is quite likely that two prototypes end up in the same data cluster, leaving one or more clusters without any prototype. To prevent this, a penalty term can be added to the objective function to push prototypes further away from each other. This works fine under certain circumstances, but it offers only an indirect influence on the repulsion behavior of the prototypes. An alternative procedure is to change the update function of the prototypes directly. Thereby, the prototypes' behavior can be easily controlled, however, at the expense of the fact, that the algorithm cannot be based on an objective function anymore.

The repulsion among the prototypes is calculated pairwise for each pair of prototypes. The strength of the repulsion depends on the distance between two prototypes and on the sum of membership values to the respective prototypes. The modified update function for repulsive prototypes is defined as

$$\hat{\beta}_i^{t+1} = \underbrace{\frac{\sum_{j=1}^{n} \left(u_{ij}^t\right)^{\omega} x_j}{\sum_{j=1}^{n} \left(u_{ij}^t\right)^{\omega}}}_{A} + c \sum_{\substack{k=1 \\ k \neq i}}^{m} \left(\underbrace{\frac{\beta_i^t - \beta_k^t}{\|\beta_i^t - \beta_k^t\|}}_{B} \underbrace{\frac{u_k^t}{u_i^t + u_k^t}}_{C} \underbrace{\varphi(\|\beta_i^t - \beta_k^t\|)}_{D} \right). \tag{3}$$

The first term (A) is identical to (2) and describes the impact of the data objects to a prototype while the rest of the formula describes the repulsion. Because the repulsion is computed for each pair of prototypes, the repulsion is calculated between prototype i and every other prototype. The term (B) is a unified vector for the direction of the repulsion. The term (C) takes the influence of the data objects to

the respective prototype into account with $u_i^t = \sum_{j=1}^{n} u_{ij}^t$. Imagine there are two prototypes in one data cluster. If they would push each other away with equal force, they would both be pushed out of the cluster. This is not the desired result, since one of them should remain inside. In crisp words: the term (C) gives that prototype more force, to which more data objects are assigned. In fuzzy terms, the prototype with the larger sum of membership values is preferred in the pairwise repulsion process. The last term (D) takes the distance between prototypes into account. The function $\varphi : \mathbb{R} \longrightarrow [0, 1]$ should be monotonically decreasing and continuous. In principle, every function that holds these constraints is valid, but here, we will consider one family of functions in particular, which is described in the next paragraph. Consider, that the value inside the sum is between 0 and 1. If the data objects are not scaled to a unified space, the influence of the repulsion might be too weak to counteract the attraction of the data objects. The parameter c handles the balance between the attraction of the data objects and the repulsion among prototypes. If the data set is standardized, c can be set to 1.

Practical tests have shown that a function of the family $\varphi(x) = \frac{1}{a(x-b)^c}$ is not suitable for the repulsion process. Instead, the logistic function turned out to be very feasible. This is why we decided to use the following variation of the logistic function:

$$\varphi(x) = \frac{1}{1 + e^{a(x-\sigma)}}.$$

The value σ is the distance at which the function φ has the value 0.5. The parameter a describes the gradient of φ at the point σ. The problem is, that such a parameter a is not very intuitive. Therefore, the definition of φ is changed to a formula that holds the two constraints: $\varphi(\sigma) = 0.5$ and $\varphi(\gamma) = \alpha$ with $\gamma > \sigma$ and $\alpha \in (0, 0.5)$. In words: the repulsion should have half its strength at a distance of σ and should have almost no effect at a distance of γ. Mathematically, "almost no effect" is described by α, so that a fixed value of $\alpha = 0.05$ might be useful. With these constraints, the parameter a can be computed by

$$a = \frac{\ln(\frac{1}{\alpha} - 1)}{\gamma - \sigma}.$$

Using the parameter σ and γ (leaving α at a fix value), it is easy and intuitive to control the repulsion. In Fig. 1, φ is plotted for several sets of parameters. Applying noise clustering with repulsive prototypes, it has proven useful to set $\sigma = 0.9\nu$, $\sigma = 1.8\nu$ and $\alpha = 0.05$ to gain well-separated clusters. Depending on the inherent data structure, appropriate values should be assigned to these parameters. ν can be interpreted as the maximal spacial extension of a cluster while σ and γ influence the minimal distance between the centers of clusters.

Tests have shown that the impact of the attracting conventional FCM-component and the repulsive component in the update equation need to be balanced appropriately. Otherwise, due to the FCM-component, prototypes will be attracted to the data clusters in one iteration followed be a strong repulsion in the next. The algorithms behavior can be described best as "nervous". To prevent this, it is necessary to define

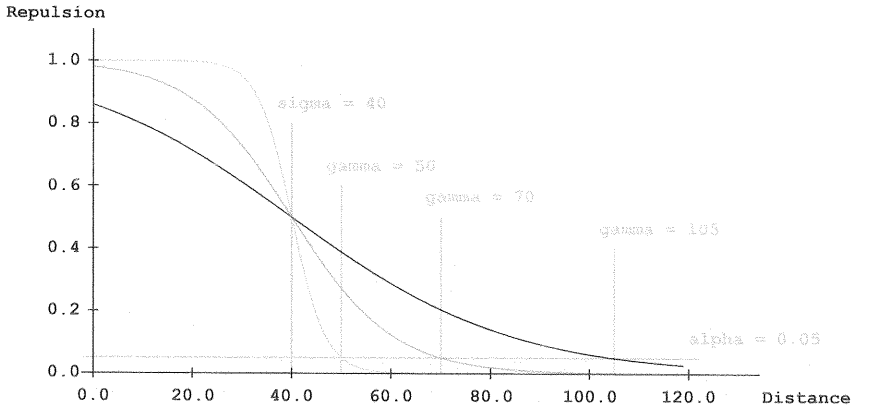

Fig. 1 Repulsion function with several parameters for γ

a relatively small learning rate $\delta \in (0, 1]$, such as $\delta = 0.1$. Then, the update formula of the prototypes is expanded to

$$\beta_i^{t+1} = \delta \hat{\beta}_i^{t+1} + (1 - \delta)\beta_i^t.$$

Sometimes, the final setup of the prototypes is not useful to generate a comprehensible partition of the data objects, because the repulsion influences the position of the prototypes so that they cannot be seen as representatives of the clusters any more. For this reason, we do not consider clustering with repulsive prototypes as an alternative to fuzzy c-means or other prototype-based clustering algorithms (Gath & Geva, 1989; Gustafson & Kessel, 1979). But the repulsion has proven very useful to solve the initially mentioned problem of finding the number and approximate position of the clusters.

Noise clustering, extended with repulsive prototypes is still a prototype-based algorithm that needs the number of prototypes to be known in advance. Since the algorithm should only be used for initialization purposes, it is not necessary to have exactly the same number of prototypes as there are clusters in a data set. Since this is the case, it is possible to overestimate the number of clusters. In fact, it is useful to overestimate the number of clusters with two or three times the prototypes as there are expected clusters in the data set. This way, it is almost guaranteed, that each cluster is found by at least one prototype. Due to the repulsive behavior of the prototypes and assuming the parameters are chosen correctly, each cluster will hold only one prototype. Accordingly, many prototypes are floating outside of clusters and might stabilize on some noise data objects. Due to the term (C) in (3), a prototype floating outside of a cluster is not able to chase away a prototype already inside of this cluster. Even if there are two prototypes initialized inside of a cluster, a small unbalance is sufficient that either of the prototypes will push out the other.

After running noise clustering with repulsive prototypes and an overestimated number of clusters, a simple test $T : \boldsymbol{B} \longrightarrow \{1, 0\}$ can be used to determine if a

prototype is considered to be inside a cluster or not. This test can depend on the specific clustering task. A very simple test would be to consider a minimal sum of membership values u_{min} of all data objects towards one prototype:

$$T(\beta_i) = \begin{cases} 1 \text{ , } \sum_{j=0}^{n} u_{ij} > u_{min}, \\ 0 \text{ , otherwise.} \end{cases}$$

Finally, the position of all positively tested prototypes can be used to initialize another prototype-based clustering algorithm such as fuzzy c-means.

Experiments have shown, that if there is little or even no noise, prototypes that are outside of data clusters do not stop moving. The reason is, that they are strongly influenced even by small changes of other prototypes. Therefore, the usual approach to terminate the algorithm, i.e., when the difference in the membership matrix from one iteration step to the next one is small ($\|U^{t-1} - U^t\| < \varepsilon$), is not applicable. A simple solution for this problem is to terminate the algorithm after a previously defined number of iterations. An alternative could be, to measure the difference of prototype positions for prototypes that are detected to be inside a cluster.

4 Experimental Results

Repulsive prototypes become of great value if there are many data sets to analyze having similar properties regarding the expected cluster size, but different number of clusters. This is the case for an actual problem in the domain of air traffic management. In this application, the airspace around airports needs to be analyzed. Groups of aircraft need to be found that approach the airport from similar directions. For this purpose, the first radar point of the aircraft inside the specified airspace is considered. When applying fuzzy c-means or noise clustering with randomly initialized prototypes, it is very unlikely that each cluster is found by exactly one prototype. The data set presented in Fig. 2 (left) is similar to one of our examples, but due to educational purposes artificially generated. An approach using noise clustering with randomly initialized prototypes often ends in a result like in Fig. 2 (right). One cluster is associated to two prototypes which results in at least one data cluster wrongly associated to the noise cluster (illustrated by the circle on the left side).

As shown in Fig. 3 (left), repulsive prototypes can be used to find the number and position of the clusters. In a second step, noise clustering can be used to partition this data set which produces the result shown in Fig. 3 (right). For this example, the following parameters were used: $\omega = 2$, $\nu = 0.2$, $m = 20$, $\sigma = 0.9\nu$, $\gamma = 1.8\nu$, $\alpha = 0.05$ and $u_{min} = 50$.

When it comes to the number of prototypes, the question may arise by what extent the number of prototypes may be overestimated. A test with artificial data has shown that there is almost no restriction to the number. Because of the quadratic nature of calculation complexity, however, it might not be wise to overestimate the number of clusters unreasonably.

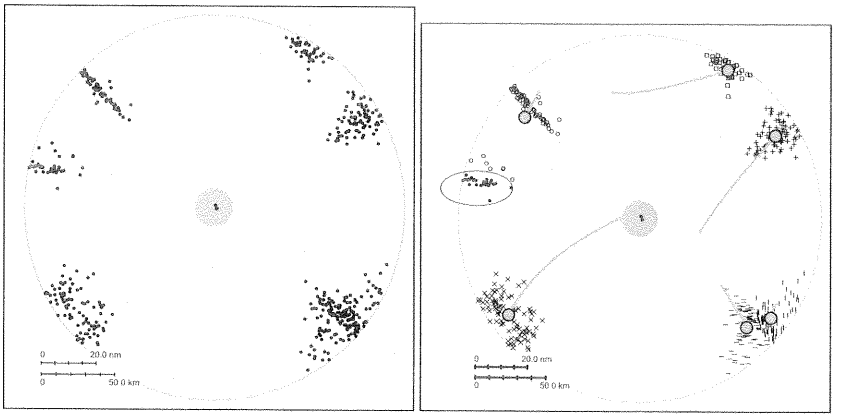

Fig. 2 Example for clustering the entrance positions of the airspace surrounding an airport. The airport is located at the *gray area in the middle*. The *big circle* has a diameter of 200 NM (370 km). The data recordings in the *middle* are considered to be noise because they are too far away from the *border*. The "tails" of the Prototypes represent their path from their initialization point to their convergence point

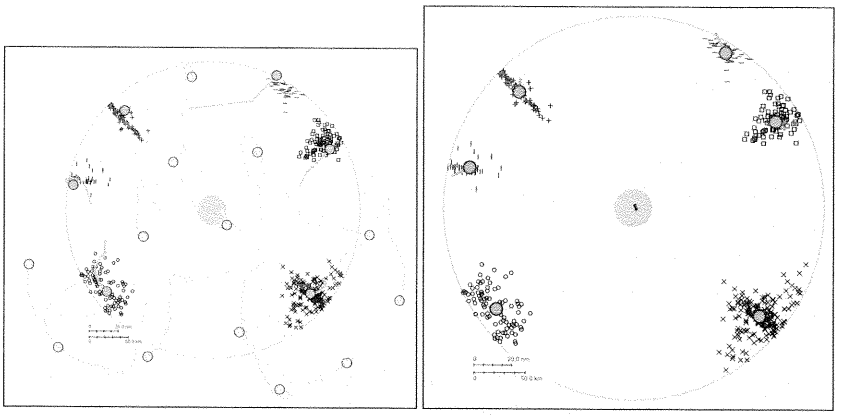

Fig. 3 Example for applying repulsive clustering on the data set of Fig. 2

As for all clustering algorithms, there are examples where repulsive prototypes do not work. An example is shown in Fig. 4. If at least one large and vast cluster is in the data set and several small in close proximity, so that the distance between the small clusters is less than the diameter of the large one, than repulsive clustering does not find a useful result. In our problem of clustering flight data, repulsive clustering worked for all examples very well. The correct number of clusters were found in all cases, only the noise distance had to be manually adjusted in some data sets due to unusual sized clusters.

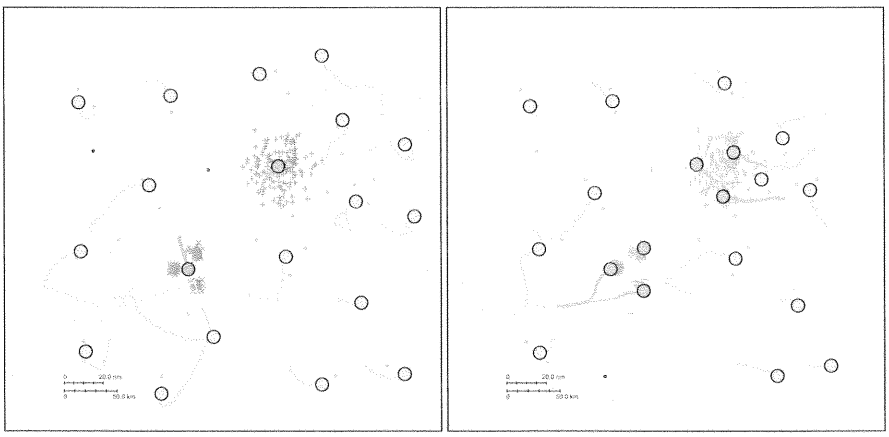

Fig. 4 Example where repulsive clustering does not produce a satisfying result. If the repulsion is strong enough so that the large clusters are represented by just one prototype, then the small clusters can not be represented by one prototype each (*left*). If the repulsion is adjusted in a way, that the small clusters are correctly approximated, the large clouds are divided into several clusters (*right*)

5 Conclusions and Future Work

We have shown that noise clustering with repulsive prototypes can be used to find the number and approximate position of clusters in a data set. This can be useful if the exact number of clusters is not known in advance or if the clusters are located in a way, that a straightforward approach with fuzzy c-means does not produce good results due to bad initialization of the prototypes. We have also shown that repulsive prototypes can be parametrized intuitively, allowing their application without expert knowledge.

The principle of repulsive prototypes allows to use them with every prototype-based clustering algorithm. Therefore, we will test the behavior of repulsive prototypes with other prototype-based clustering algorithms. We have not tested the behavior in high dimensional spaces, which will be done in near future. The problem of applying repulsive prototypes on problematic data sets like shown in the last section, might be solved by localized distance measures.

References

Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms.* New York: Plenum.

Dave, R. N., & Krishnapuram, R. (1997). Robust clustering methods: a unified view. *IEEE Transactions on Fuzzy Systems, 5*, 270–293.

Gath I., & Geva, A. B. (1989). Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 11*, 773–781.

Gustafson, D. E., & Kessel, W. C. (1979). *Fuzzy clustering with a fuzzy covariance matrix.* In Proceedings of the IEEE Conference on Decision and Control, San Diego, 761–766.