FUZZY CLUSTERING OF DOCUMENTS

Matjaž Juršič, Nada Lavrač
Department of Knowledge Discovery
Jozef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel: +386 1 4773125
E-mail: *matjaz.jursic@ijs.si*

ABSTRACT

This paper presents a short overview of methods for fuzzy clustering and states desired properties for an optimal fuzzy document clustering algorithm. Based on these criteria we chose one of the fuzzy clustering most prominent methods – the c-means, more precisely probabilistic c-means. This algorithm is presented in more detail along with some empirical results of the clustering of 2-dimensional points and documents. For the needs of documents clustering we implemented fuzzy c-means in the TextGarden environment. We show few difficulties with the implementation and their possible solutions. As a conclusion we also propose further work that would be needed in order to fully exploit the power of fuzzy document clustering in TextGarden.

1 INTRODUCTION

Clustering is an unsupervised classification of objects (data instances) into different groups. In particular we are talking about the partitioning of a dataset into subsets (clusters), so that the data in each subset (ideally) share some common property. This property is usually defined as proximity according to some predefined distance measure. The goal is to divide the dataset in such a way that objects belonging to the same cluster are as similar as possible, whereas objects belonging to different clusters are as dissimilar as possible. The computational task of classifying the data set into k clusters is often referred to as k-clustering. Although estimating the actual number of clusters (k) is an important issue we leave it untouched in this work.

Fuzzy clustering [1, 2] in contrast to the usual (crisp) methods does not provide hard clusters, but returns a degree of membership of each object to all the clusters. The interpretation of these degrees is then left to the user that can apply some kind of a thresholding to generate hard clusters or use these soft degrees directly.

All the algorithms that we consider here are partitional, deterministic and non-incremental (based on the taxonomy defined in [4]). The property that we want to change using fuzzy methods instead of crisp clustering is exclusiveness, as there are cases in which algorithms constructing overlapping partitions of set of documents perform better than the exclusive algorithms.

Text-Garden [3] is a software library and collection of software tools for solving large scale tasks dealing with structured, semi-structured and unstructured data – the emphasis of its functionality is on dealing with text. It can be used in various ways covering research and applicative scenarios. Our special interest in TextGarden is the OntoGen tool [7]. Ontogen is a semi-automated, data-driven ontology construction tool, focused on the construction and editing of topic ontologies based on document clustering. Actually we want to upgrade OntoGen with fuzzy clustering properties; however, since it is based on TextGarden we must provide the implementation of the fuzzy clustering algorithm in its library.

2 FUZZY CLUSTERING ALGORITHMS

In this section, we present some of the fuzzy clustering algorithms mainly based on the descriptions in [5]. We devote the majority of space to the hard c-means, fuzzy c-means and possibilistic c-means. For the other methods we provide just a short description, as we did not find them appropriate for our needs.

All algorithms described here are based on objective functions, which are mathematical criteria that quantify the quality of cluster models. The goal of each clustering algorithm is the minimization of its objective function. The following syntax will be used in the equations, algorithms and their explanations:

J ... objective function

$\mathrm{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_\mathrm{n}\}$ ... dataset of all objects (data instances)

$\mathrm{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_\mathrm{c}\}$ ... set of cluster prototypes (centroid vectors)

$d_{ij} = \|\mathbf{x}_1 - \mathbf{c}_1\|$... distance between object $\mathbf{x}_\mathrm{j}$ and centre $\mathbf{c}_\mathrm{i}$

u_{ij}... weight of assignment of object $\mathbf{x}_\mathrm{j}$ to cluster i

$\mathbf{u}_j = (u_{1j}, \ldots, u_{cj})^T$ ... memberships vector of object $\mathbf{x}_\mathrm{j}$

$\mathrm{U} = (u_{ij}) = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$ ... partition matrix of size $c \times n$

2.1 Hard c-means (HCM)

Hard c-means is better known as k-means and in general this is not a fuzzy algorithm. However, its overall structure is the basis for all the others methods. Therefore we call it hard c-

means in order to emphasize that it serves as a starting point for the fuzzy extensions.

The objective function of HCM can be written as follows:

$$J_h = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} d_{ij}^2. \quad (2.1)$$

As mentioned HCM is a crisp algorithm, therefore: $u_{ij} \in \{0,1\}$. It is also required that each object belongs to exactly one cluster: $\sum_{i=1}^{c} u_{ij} = 1, \forall j \in \{1, \dots, n\}$.

Before outlining the algorithm, we must know how to calculate new membership weights:

$$u_{ij} = \begin{cases} 1, & \text{if } i = \operatorname{argmin}_{l=1}^{c} d_{lj} \\ 0, & \text{otherwise} \end{cases}, \quad (2.2)$$

and based on the weights, how to derive new cluster centres:

$$\mathbf{c}_i = \frac{\sum_{j=1}^{n} u_{ij} \mathbf{x}_j}{\sum_{j=1}^{n} u_{ij}} \quad (2.3)$$

The algorithm can now be stated very simply as shown in Table 2.1.

INPUT: A set of learning objects to be clustered and the number of desired clusters *c*

OUTPUT: Partition of learning examples into *c* clusters and membership values u_{ij} for each example $\mathbf{x}_j$ and cluster *i*.

ALGORITHM (2.1) ***The hard c-means algorithm*:**

```
(randomly) generate clusters centres
repeat
    for each object recalculate membership weights
    using equation (2.2)
    recompute the new centres using equation (2.3)
until no change in C can be observed
```

Table 2.1: Pseudocode of the HCM clustering algorithm.

The HCM algorithm has a tendency to get stuck in a local minimum, which makes it necessary to conduct several runs of the algorithm with different initializations. Then the best result out of many clusterings can be chosen based on the objective function value.

2.2 Fuzzy c-means (FCM)

Probabilistic fuzzy cluster analysis [1, 2] relaxes the requirement: $u_{ij} \in \{0,1\}$, which now becomes: $u_{ij} \in [0,1]$. However $\sum_{i=1}^{c} u_{ij} = 1, \forall j \in \{1, \dots, n\}$ still holds. FCM optimizes the following objective function:

$$J_f = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2. \quad (2.4)$$

Parameter *m*, $m>1$, is called the fuzzyfier or the weighting exponent. The actual value of *m* determines the 'fuzziness' of the classification. It has been shown [5] that for the case $m=1$, J_f becomes identical to J_h and thus FCM becomes identical to hard c-means.

The transformation from the hard c-means to the FCM is very straightforward; we must just change the equation for calculating memberships (2.2) with:

$$u_{ij} = \frac{1}{\sum_{l=1}^{c} \left(\frac{d_{ij}^2}{d_{lj}^2} \right)^{\frac{1}{m-1}}} = \frac{d_{ij}^{\frac{-2}{m-1}}}{\sum_{l=1}^{c} d_{lj}^{\frac{-2}{m-1}}}, \quad (2.5)$$

and function for recomputing clusters centres (2.3) with:

$$\mathbf{c}_i = \frac{\sum_{j=1}^{n} u_{ij}^m \mathbf{x}_j}{\sum_{j=1}^{n} u_{ij}^m}. \quad (2.6)$$

Equation (2.5) clearly shows the relative character of the probabilistic membership degree. It depends not only on the distance of the object $\mathbf{x}_j$ to the cluster $\mathbf{c}_i$, but also on the distances between this object and other clusters.

Although the algorithm stays the same as in HCM (Table 2.1), we get probabilistic outputs if we apply above changes. The (probabilistic) fuzzy c-means algorithm is known as a stable and robust classification method. Compared with the hard c-means it is quite insensitive to its initialization and it is not likely to get stuck in an undesired local minimum of its objective function in practice. Due to its simplicity and low computational demands, the probabilistic FCM is a widely used initializer for other more sophisticated clustering methods.

2.3 Possibilistic c-means (PCM)

Although often desirable, the relative property of the probabilistic membership degrees can be misleading. High values for the membership of object in more than one cluster can lead to the impression that the object is typical for the clusters, but this is not always the case. Consider, for example, the simple case of two clusters shown in Figure 2.1. Object $\mathbf{x}_1$ has the same distance to both clusters and thus it is assigned a membership degree of about 0.5. This is plausible. However, the same degrees of membership are assigned to object $\mathbf{x}_2$ even though this object is further away from both clusters and should be considered less typical. Because of the normalization the sum of the memberships has to be 1. Consequently $\mathbf{x}_2$ receives fairly high membership degrees to both clusters. For a correct interpretation of these memberships one has to keep in mind that they are rather degrees of sharing than of typicality, since the constant weight of 1, given to an object, must be distributed over the clusters.

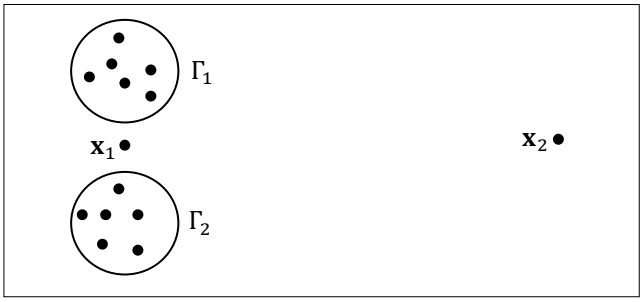

Figure 2.1: Example of misleading interpretation of the FCM membership degree.

Therefore PCM, besides relaxing the condition for u_{ij} to $u_{ij} \in [0,1]$ as in case of FCM, also drops the normalization requirement: $\sum_{i=1}^{c} u_{ij} = 1, \forall j \in \{1, \dots, n\}$. The probabilistic objective function J_f that just minimizes squared distances would be inappropriate because with dropping of the normalization constraint a trivial solution exists for $u_{ij} = 0$ for all $i \in \{1, \dots, c\}$ and $j \in \{1, \dots, n\}$, i.e., all clusters are empty. In order to avoid this solution, penalty a term is introduced that forces the memberships away from zero. Objective function J_f is modified to:

$$J_p = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} (1 - u_{ij})^m, \quad (2.7)$$

where $\eta_i > 0$ for all $i \in \{1, \dots, c\}$.

In the PCM algorithm, the equation for calculating cluster centres stays the same as in FCM (2.6). But the equation for recalculating membership degrees changes from (2.5) to:

$$u_{ij} = \frac{1}{\left(\frac{d_{ij}^2}{\eta_i}\right)^{\frac{1}{m-1}}}. \quad (2.8)$$

This also slightly changes the original procedure (Table 2.1) since we must recompute η_i using the equation (2.9) before calculating the weight u_{ij}.

$$\eta_i = \frac{\sum_{j=1}^{n} u_{ij}^m d_{ij}^2}{\sum_{j=1}^{n} u_{ij}^m}. \quad (2.9)$$

Properties of PCM [5] are the following:

- Cluster Coincidence: since PCM is not forced to partition data exhaustively it can lead to solutions where two or more clusters occupy the same space (same objects with the same membership weighting).
- Cluster Repulsion: objective function J_p is, in general, fully optimized only if all clustered centres are identical. Because of that, other, not optimal solutions are found just as a side effect of J_p getting stuck in a local optimum.

Because of these unwanted properties we did not choose PCM to be our choice for the implementation. However we also did not decide that this method is totally inappropriate for us. Thus we leave this matter open as the future possibility of implementing PCM in TextGarden library.

2.4 Other reviewed algorithm

During the review of fuzzy clustering algorithms we considered also the following algorithms. We will not precisely describe them in this paper, since we decided that they are not the best choice for our implementation. An interesting reader can find their descriptions in [6] or [5].

- Gustafson-Kessel Algorithm: while FCM and PCM can only detect spherical clusters GKA can identify also clusters of different forms and sizes. It is more sensitive to initialization and has higher computational costs.
- Fuzzy Shell Clustering: can, in contrast to all the algorithms above, identify also non-convex shaped clusters. They are especially useful in the area of image recognition. We think that this property in not needed in text clustering.
- Kernel-based Fuzzy Clustering: are variants of fuzzy clustering algorithms that modify the distance function to handle non-vectorial data, such as sequences, trees or graphs, without the requirement to completely modify the algorithm itself. In text clustering we are dealing with vectors so there is no need for such an advanced method.

3 IMPLEMENTATION

3.1 Evaluation on 2-dimensional points

Before having implemented FCM in the TextGarden environment we tested the algorithm on 2-dimensional points. Data was generated artificially using normally distributed clusters of random size, position and standard deviation. Empirical evaluations showed us some of the advantages of FCM compared to hard c-means:

- Lower probability of getting caught in the local optimum. We found few test scenarios where HCM gets stuck in local optima in approximately 50% of all runs but FCM never, using the same initial distributions. We could not find example where FCM would provide a non-optimal solution, but it should be noted that we knew and used the correct number of clusters c for both algorithms.
- Better correct centre (centroid vector) localization (at least on the normally distributed artificial data).

The main reason against using FCM is its higher computational complexity.

3.2 Definition of a distance measure

One of the problems that we encountered during the implementation was how to define a measure of distance between objects (or between an object and a centre of clusters). TextGarden library uses mainly a measure of similarity based on the cosine similarity. This proximity measure ranges from 0 to 1 where 0 means no similarity and 1 means total equality of vectors:

$$sim(\mathbf{x}_1, \mathbf{x}_2) = \cos\theta = \frac{\mathbf{x}_1 \cdot \mathbf{x}_2}{\|\mathbf{x}_1\|\|\mathbf{x}_2\|} \in [0,1], \quad (3.1)$$

where $\mathbf{x}_i$ is an object or more specifically in our case a bag-of-word vector representation of a document and θ is $\measuredangle(\mathbf{x}_1, \mathbf{x}_2)$. Our problem was that we actually needed the opposite of the similarity – a distance for the FCM algorithm. The two most obvious ways how to derive a distance are:

$$dist(\mathbf{x}_1, \mathbf{x}_2) = 1 - sim(\mathbf{x}_1, \mathbf{x}_2) \in [1,0], \quad (3.2)$$

$$dist(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{sim(\mathbf{x}_1, \mathbf{x}_2)} \in [\propto, 1], \quad (3.3)$$

The difficulty of (3.2) is that it's not preserving relations i.e. if $\mathbf{x}_1$ is two times more similar to $\mathbf{c}$ than $\mathbf{x}_2$ it is not necessary

that $\mathbf{x}_1$ will be also two times closer to $\mathbf{c}$ than $\mathbf{x}_2$. On the other hand (3.3) has another unwanted property. Its image interval starts from 1 and not from 0 as we would like to have if vectors are equal.

We tried both distances and evaluated them also experimentally. We have not discovered any significant change in FCM behaviour regardless of the selected distance. Thus we decided for (3.2) because it is simpler for calculation and we do not need to check for infinite numbers which results in faster execution.

3.3 Time complexity

Time complexities of HCM and FCM are respectively:

$$O_{HCM} = O(i_{HCM} \cdot n \cdot c \cdot v), \tag{3.4}$$

$$O_{FCM} = O\big(i_{FCM} \cdot n \cdot c \cdot (v + c)\big), \tag{3.5}$$

where i is the number of required iterations and v is the length of an example vector. According to our experimental results i_{FCM} is slightly higher than i_{HCM}. Consequently we assume that they share the same order of magnitude and are therefore equal as this analysis is concerned.

We can declare that the statement:

$$O(i \cdot n \cdot c \cdot v) \sim O\big(i \cdot n \cdot c \cdot (v + c)\big) \tag{3.6}$$

holds if dimensionality of the vector v is much higher than the number of clusters c. This is also the case for text clustering in TextGarden, so we can confirm that the time complexity of fuzzy c-means is similar to the one of hard c-means. Certainly we must admit that there is probably some constant factor linking the actual speeds because of the higher complexity of the inner most loops (calculation of distances and weights) of FCM compared to HCM. We estimate this factor to be in the range from 1 to 3.

3.4 An experiment on the documents data

Table 3.1 shows the results of documents clustering for both algorithms (FCM and HCM). As a set of documents we used 1000 random texts from the Yahoo Finance dataset of the companies' descriptions. We partitioned the set into 5 clusters using the same initial distributions and the same shared parameters. For each cluster we provide the mean inner similarity value, the number of documents and the three most characteristic keywords. The clusters are aligned therefore the results can be directly compared. It is evident that both algorithms found similar clusters. The average mean similarity is lower for c-means which might be the result of better centre localization of c-means.

Documents: 1000 (FCM)	Documents: 1000 (HCM)
Mean Similarity: 0.182	Mean Similarity: 0.177
Mean Sim.0.443, 92 Docs.	Mean Sim.0.369, 124 Docs.
'BANKING':0.854	'BANKING':0.770
'LOANS':0.254	'INSURANCE':0.404
'DEPOSITS':0.113	'LOANS':0.166
Mean Sim.0.137, 269 Docs.	Mean Sim.0.145, 218 Docs.
'GAS':0.247	'GAS':0.263
'EXPLORATION':0.240	'POWER':0.244
'PROPERTY':0.180	'EXPLORATION':0.199
Mean Sim.0.180, 180 Docs.	Mean Sim.0.181, 170 Docs.
'DRUGS':0.386	'DRUGS':0.386
'PHARMACEUTICALS':0.260	'PHARMACEUTICALS':0.263
'DISEASES':0.229	'CHEMICALS':0.245
Mean Sim.0.244, 107 Docs.	Mean Sim.0.155, 187 Docs.
'INSURANCE':0.623	'PROPERTY':0.303
'INVESTMENTS':0.261	'INVESTMENTS':0.271
'INSURANCE_COMPANY':0.173	'SECURITIES':0.191
Mean Sim.0.129, 352 Docs.	Mean Sim.0.134, 301 Docs.
'WIRELESS':0.202	'SOLUTIONS':0.203
'SOLUTIONS':0.181	'STORES':0.191
'SOFTWARE':0.175	'SOFTWARE':0.181

Table 3.1: Comparison of HCM and FCM algorithms on the Yahoo Finance dataset

4 CONCLUSIONS

This paper presents an overview of fuzzy clustering algorithms that could be potentially suitable for document clustering, a new fuzzy c-means clustering algorithm implemented in the TextGarden environment, and an empirical comparison of hard c-means and fuzzy c-means as an application on documents and 2D points.

Further work will consider: connecting fuzzy c-means with Ontogen and designing and implementing some adaptive threshold approach for converting fuzzy cluster to its crisp equivalent. This should be done in such a way that one document could be assigned to none, one or more clusters according to its membership degrees and similarities to the clusters. Furthermore we will perform statistical evaluation of hard c-means and fuzzy c-means in terms of document classification using other quality measures (besides average similarity) for generated clusters.

REFERENCES

[1] Dunn, J., C., A Fuzzy Relative of the ISODATA Process and its Use in Detecting Compact Well-Separated Clusters, *Journal of Cybernetics 3,* pp. 32-57, 1973.

[2] Bezdek, J., C., Pattern Recognition with Fuzzy Objective Function Algoritms, *Plenum Press, New York*, 1988.

[3] TextGarden - Text-Mining Software Tools. Available online at http://kt.ijs.si/dunja/TextGarden/.

[4] Kononenko, I., Kukar, M., Machine Learning and Data Mining: Introduction to Principles and Algorithms, *Horwood Publishing*, pp 312-358, 2007.

[5] Valente de Oliveira, J., Pedrycz, W., Advances in Fuzzy Clustering and its Applications, *John Wiley & Sons*, pp 3-30, 2007.

[6] Höppner, F., Klawonn, F., Krise, R., Runkler, T., Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition, *John Wiley & Sons*, pp 5-114, 2000.

[7] Fortuna, B., Mladenić, D., Grobelnik, M. Semiautomatic construction of topic ontologies. In: Ackermann et al. (eds.) Semantics, Web and Mining. LNCS (LNAI), *Springer* , vol. 4289, pp. 121–131., 2006.