Multi-objective Clustering Algorithm with Parallel Games

Dalila Kessira  
Laboratory of Medical Informatics (LIMED)  
University of Bejaia  
Bejaia, Algeria  
dalila.kessira@univ-bejaia.dz

Mohand-Tahar Kechadi  
Insight Center for Data Analytics  
University College of Dublin  
Dublin, Ireland  
tahar.kechadi@ucd.ie

Abstract—Data mining and knowledge discovery are two important growing research fields in the last few decades due to the abundance of data collected from various sources. The exponentially growing volumes of generated data urge the development of several mining techniques to feed the needs for automatically derived knowledge. Clustering analysis (finding similar groups of data) is a well-established and widely used approach in data mining and knowledge discovery. In this paper, we introduce a clustering technique that uses game theory models to tackle multi-objective application problems. The main idea is to exploit a specific type of simultaneous move games, called congestion games. Congestion games offer numerous advantages ranging from being succinctly represented to possessing a Nash equilibrium that is reachable in a polynomial-time. The proposed algorithm has three main steps: 1) it starts by identifying the initial players (or the cluster-heads); 2) then, it establishes the initial clusters’ composition by constructing the game to play and try to find the equilibrium of the game. The third step consists of merging close clusters to obtain the final clusters. The experiment results show that the proposed clustering approach obtains good results and it is very promising in terms of scalability, and performance.

Keywords—Data mining, data analysis, clustering, game theory, simultaneous-move games, Nash equilibrium.

I. INTRODUCTION

Data science is one of the most growing research fields over the last few years. It refers to an empirical approach that uses the available big amounts of data to provide answers to a wide variety of questions and problems that human beings are enable to treat without the intervention of the machine [1]. Among the several methods and techniques used in data science, cluster analysis (clustering) is a well-established and wide-used technique. It is the process of grouping the data into classes or clusters, such a way that objects in the same cluster are very similar and objects in different clusters are different.

Game theory offers very attractive rigorous mathematical tools for modeling and resolving a wide part of strategic situations, thus it is increasingly used to resolve a wide range of problems.

Game theory-based clustering algorithms are not numerous; however, However, only a few do reach the Nash equilibrium in a polynomial time. This is due to the big complexity of finding solution concepts, such as Nash equilibrium[2]. There are many other types of clustering algorithms based on the various methodologies used (see [3][4], [5], for more details).

In this paper, we propose a multi-objective clustering algorithm based on game theory using parallel games named MOCA-SM (Multi-Objective Clustering Algorithm based on Simultaneous-Move games).

The rest of this paper is organized as follows: in section 2, we give an overview of Game Theory and its basic notions. In section 3 we present the proposed clustering approach. In Section 4 we provide the experimental results and their discussion. In Section 5 we conclude this paper and we give some perspectives.

II. BACKGROUND

In this section, we first introduce the game theory, then, we present the sub-class of games used in this work named Singleton Congestion games with player-specific payoff functions.

A. Parallel games

In game theory, the term “game” means an abstract mathematical model of a multi-agent decision-making setting.[6]

Formally, it is represented [7] by the tuple (N, S, u):  
- N = {1, ..., n} a set of players, indexed by i;  
- S=S*1*...*S*n, where Si is a finite set of strategies available to player i. Each vector s= (s1, ..., sn) ∈ S is called a strategy profile;  
- u= (u1, ..., un), where ui: S−→ R is a real-valued utility (or payoff) function for player i.

The normal form, also known as the strategic or matrix form is when we represent a game via an n-dimensional matrix. Simultaneous-move (parallel) games [8] are when all players take their decisions at the same time, consequently, they don’t know the decisions made by the other players.

B. Solution concepts and Nash equilibrium

Reasoning about multi-player games is based on solution concepts, which are principals that help us to identify interesting subsets of the outcomes of a game [7]. One of the most powerful solution concepts in game theory is Nash equilibrium.

To define Nash equilibrium [2], we need to define the notion of best response [7]. Formally, we denote s−i = (s1, ..., si−1, si, si+1, ..., sn), a strategy profile s without player i’s strategy. Thus we can write s = (si, s−i). If the agents other than i (whom we denote −i) were to commit to playing s−i, a utility-maximizing agent i would face the problem of determining his best response.
Definition 1 (Best response) [7]: Player i’s best response to the strategy profile \( s_i \) is a mixed strategy \( s_i^* \in S_i \) such that \( u_i(s_i, s_{-i}) \geq u_i(s_i^*, s_{-i}) \) for all strategies \( s_i \in S_i \).

The best response is not necessarily unique. Thus, the notion of best response is not a solution concept; it does not identify an interesting set of outcomes in this general case [7].

Definition 2 (Nash equilibrium) [2]: A strategy profile \( s = (s_1, \ldots, s_n) \) is a Nash equilibrium if, for all agents \( i, s_i \) is a best response to \( s_{-i} \).

Nash equilibrium is a simple but powerful principle for reasoning about behavior in general games [2]: even when there is no dominant strategy, we should expect players to use strategies that are best responses to each other [9]. This is the most used solution concept in game theory, nevertheless, it is established that computing Nash equilibrium is hard for a lot of games’ sub-classes [10]–[13].

C. Singleton Congestion games with player-specific payoff functions

Milchtaich [14] in 1996, introduced a sub-class of congestion games presented by Rosenthal [15]. In this sub-class, the payoff function associated with each resource is not universal but player-specific. And on the other hand, it is assumed that each player chooses only one primary resource \( n \) and that the payoff received actually decreases (not necessarily strictly so) with the number of other players selecting the same resource.

Consequently, the cost of a resource \( e \) for player \( i \), will not depend on the number of players that chose the resource, but it will depend also on the player himself. Hence, every player \( i \) has its cost function \( c_{i,e} \) for every resource \( e \), hence the definition below.

Definition 3 (SCGPSC: Singleton congestion game with player-specific cost [14]) A SCGPSC game is the tuple \( (N, E, S, (c_{e,i})_{e \in N, i \in N}) \):

- \( N = \{1, \ldots, n\} \) a set of players, indexed by \( i \).
- \( E = \{1, \ldots, m\} \) a set of resources, indexed by \( e \).
- Strategy set \( S: S = S_1 \times \cdots \times S_n \) where \( S_i \) is a set of strategies available to agent \( i \) where strategy \( s_i \in S_i \) is a singleton subset of resources, i.e. set with exactly one element.

- Cost function \( c_{e,i}(n_e) \in \mathbb{R} \) for player \( i \) and resource \( e \), where it can be \( c_{e,i}(n_e) \neq c_{e,i}(n_e) \) if \((i, i') \in N \times N\), i.e. the cost function depends on the player himself and on the number of players that select the resource \( e \).

Where: \( n_e = \{|i: e = s_i\|\} \), and \( n_1, n_2, \ldots, n_m \) is called the congestion vector corresponding to a strategy \( s = (s_1, s_2, \ldots, s_n) \).

This class of congestion games will be used in our approach to model and resolve the clustering problem.

Nash equilibrium in SCGPSC

Malchtaich [14] proved that for this sub-class of congestion games doesn’t generally admit a potential [15], so, best response dynamics don’t always converge to Nash equilibrium as it may be cyclic.

However, Malchtaich proved that SCGPSC possess always a Nash equilibrium. He used the proof by induction on the number \( n \) of players, where it is supposed that an instance of the game with \( n-1 \) players has a Nash equilibrium and it’s proved that the game with \( n \) players has a Nash equilibrium. For the complete proof see [14].

This existence proof is constructive and implicitly describes an efficient algorithm for finding an equilibrium in a given \( n \)-player SCGPSC, by adding one player after the other in at most \( \binom{n+1}{2} \) steps [14].

III. PROPOSED APPROACH

Our main contribution is the design of a multi-objective clustering algorithm based on game theory with parallel (or simultaneous-move) games. We use the class of non-cooperative simultaneous-move game presented above, named Singleton Congestion game with Player-Specific Cost (SCGPSC) to model the clustering problem, where players are a subset of the initial dataset, the resources are the rest of the dataset, and the cost function is an optimization function of two contradictory objectives: connectedness and separation.

A. Optimization objectives

Our approach is based on optimizing two conflicting optimization objectives, the first is R-Square, and the other is the connectivity of the clusters based on the Euclidean distance. The combination of those two objectives was first used by Heloulu et al. [16] in a clustering algorithm that uses sequential game theory. R-Square [17], [18] is optimal when the number of clusters is high and the connectivity of clusters is optimal when the number of clusters is low, so, the compromise of those two conflicting objectives guarantees a good quality clustering according to experiments conducted by [16].

R-square is given by the formula:

\[ R^2(C) = \frac{I_g(C)}{I_g(C) + I_e(C)} \]  \hspace{1cm} (1)

Where \( I_g(C) \) is inter-cluster inertia which measures the separation of the clusters [19]:

\[ I_g(C) = \frac{1}{n} \sum_{i=1}^{K} |C_i| \cdot d^2(\chi_i, g_i) \] \hspace{1cm} (2)

Where \( \chi_i \) is the cluster-head of cluster \( C_i \), \( g_i = (g_i, \ldots, g_i) \) and \( g_i \) is the gravity center of the dataset along the \( j \)th dimension.

\[ g_j = \frac{1}{n} \sum_{i=1}^{n} a_{ij} \] \hspace{1cm} (3)

\[ I_g(C) \] is the intra-cluster inertia which should be as weak as possible to have a set of homogeneous clusters. It is given as [19]:

\[ I_g(C) = \frac{1}{n} \sum_{i=1}^{K} \sum_{c_{i,c} \in C_i} d^2(\chi_i, c_{i,c}) \] \hspace{1cm} (4)

Where: \( c_{i,c} \) is the cluster-head of the cluster \( C_i \).

The connectivity measure is intended to assign similar data into the same cluster. It is given by the formula:

\[ \text{Connect}(C_i) = \frac{\sum_{c_{i,c} \in C_i} \gamma_{i,c}}{n} \] \hspace{1cm} (5)

Where \text{Connect}(C_i) is the connectivity of the cluster \( C_i \), and it is given by the formula:

\[ \text{Connect}(C_i) = \frac{1}{|C_i|} \sum_{t=1}^{n} \sum_{c_{i,c} \in C_i} X_{r,s} \] \hspace{1cm} (6)

Where:

\[ X_{r,s} = \begin{cases} 1, & \text{if } r,s \in C_i \\ 0, & \text{otherwise} \end{cases} \]
**Algorithm 1: MOCA-SM**

**Inputs:** Dataset \( O=\{o_0, o_1, \ldots, o_{m-1}\} \), number of final clusters \( f \)

**Outputs:** set of clusters \( C \)

1: \( t \leftarrow 0 \)
2: compute distance matrix \( dis \) between all objects of \( O \)
3: **Algorithm 2:** Identification of initial players, the set \( ch \)
4: **repeat**
5: \( t \leftarrow t + 1 \)
6: **Algorithm 3:** construct the game in formula (8)
7: **Algorithm 4:** compute Nash equilibrium
8: **for each** player
9: if Nash equilibrium enhance the clustering then
10: Allocate the Nash equilibrium strategy for player
11: else
12: Player out of game
13: **end if**
14: **end for**
15: until \( O \) is empty or all players are out of the game
16: **Algorithm 5:** Merge the resulted clusters
17: assign the left objects to the closest cluster

**Algorithm 2: Identification of initial players**

**Inputs:** Dataset \( O=\{o_0, o_1, \ldots, o_{m-1}\} \)

**Outputs:** Set \( ch \) of initial cluster-heads (players) \( ch=\{ch_1, ch_2, \ldots, ch_k\} \)

1: compute \( n_0 \), the initial number of players using formula (9)
2: compute dissimilarity of each object \( o \) of \( O \): \( dm(o) \) using formula (10)
3: \( O' \leftarrow O \)
4: **repeat**
5: Compute the object \( o_e \) with the most density around in \( O' \)
6: Add \( o_e \) to \( ch \)
7: Eliminate \( o_e \) from the \( O' \)
8: Eliminate close objects to \( o_e \), from the \( O' \)
9: until \( O' \) is empty or \( |ch| = n_0 \)

**Algorithm 3: Formulation of the game**

**Inputs:** players \( ch=\{ch_1, ch_2, \ldots, ch_k\} \), dataset \( O=\{o_0, o_1, \ldots, o_j\} \)

**Outputs:** Matrix \( Cost \) containing the costs of all strategies to all players

1: Identify available strategies over the set \( O \)
2: **for each** player \( e \) in \( ch \)
3: **for each** strategy
4: **for each** value of the congestion vector
5: compute the cost of the singleton strategy for the player using formula
6: assign the cost to the matrix \( Cost \)
7: **end for**
8: **end for**
9: **end for**

**Algorithm 4: Merging close clusters**

**Inputs:** set of initial clusters set \( C=\{c_0, c_1, \ldots, c_{m0-1}\} \), the number of the final clusters \( f \)

**Outputs:** set of final clusters \( C \)

1: compute the distance between all initial clusters
2: **repeat**
3: merge the two closest clusters
4: **until the number of clusters is reached**
TABLE 1. DATASETS DESCRIPTION

| Datasets                  | Instances | Attributes | Clusters |
|--------------------------|-----------|------------|----------|
| Synthetic datasets       |           |            |          |
| Spharical_3_4            | 400       | 3          | {100, 100, 100, 100} |
| Dataset_3_2              | 76        | 2          | {13, 43, 20} |
| Spiralsquare             | 1500      | 2          | {116, 134, 125, 125, 500, 500} |
| Real-world datasets      |           |            |          |
| Iris                     | 150       | 4          | {50, 50, 50} |
| Wine                     | 178       | 13         | {59, 71, 48} |

Fig. 1. Synthetic datasets, (a) Spherical_3_4 (b) Dataset_3_2 (c) Spiralsquare

Step 1: Identification of the initial players

This step aims at the identification of the players (cluster heads). The initial number of clusters is estimated using the formula:

\[ n_0 = m/L \]  

(9)

Where: \( m \) is the size of the dataset, and \( L \) is the number of neighbors to use to find the connectivity of each cluster. After that, the dissimilarity of each object compared to all other objects in the dataset is computed using this formula:

\[ d_m(o) = \frac{1}{m-1} \sum_{o \in O, o \neq o} d[s[o][o_i] \]  

(10)

Objects with the smallest value of \( d_m \) are the objects with the most density around them; consequently, they are chosen to be cluster-heads (or players) to play over the other objects.

Step 2: establishment of initial clusters’ composition

This step aims at determining the composition of initial clusters by constructing the game described in formula (8). The players of the game are the objects with the highest density around them; this game is played several times until all objects are allocated or when no player wants to play again. Algorithm 3 summarizes this step.

Step 3: Merging close clusters

After having \( n_0 \) initial clusters from step 2, this step is about merging them until we have the exact number of clusters. First, the distance between each pair of the initial clusters is calculated, then, the most two closest clusters are merged until we reach the desired number of clusters.

IV. EXPERIMENTATIONS AND CLUSTERING VALIDATION

A. Settings

Experiments were conducted on 2.50GHz Intel® Core™ i5-3210M with 8Go RAM. All experiments were implemented in Java programming language.

We compared our algorithm with well-known clustering algorithms, K-means, DBSCAN (density-based spatial clustering of applications with noise), and SOM (Self-organizing Map). The source code for those algorithms is used from Java Machine Learning Library JavaML 1. As for K-means and SOM, the obtained results are the mean of 10 runs.

The parameter \( L \) is the number of neighbors taken into consideration when calculating the connectivity. In our experiments’ setting, we used the value 9 when the size of the dataset is less than 150; the value 14 when the size of the dataset is between 150 and 500; the value 28 when the dataset is bigger than 500.

Experiments were conducted on five datasets, three of them are synthetic: Spherical_3_4, Dataset_3_2, and Spiralsquare, the other two are real-world: Iris and Wine which are well-known datasets for cluster evaluation.

1 http://java-ml.sourceforge.net/

2 Available at GitHub: https://github.com/deric/clustering-benchmark/tree/b47c8b77028b61d632c2c3901868e99444b350

3 Available at GitHub: https://github.com/deric/clustering-benchmark/tree/b47c8b77028b61d632c2c3901868e99444b350

4 Available at UCI Machine Learning Repository: ftp://ftp.ics.uci.edu/pub/machine-learning-databases

5 Available at UCI Machine Learning Repository: ftp://ftp.ics.uci.edu/pub/machine-learning-databases
Fig. 2. Purity values resulted from various Algorithms on different datasets. A high value of purity indicates a better clustering result.

Fig. 3. Rand Index resulted from running various Algorithms on different datasets. A high value of Rand Index indicates a better clustering result.

Fig. 4. F-measure resulted from running various Algorithms on different datasets. A high value of F-measure indicates a better clustering result.

Fig. 5. ARI resulted from running various Algorithms on different datasets. A high value of ARI indicates a better clustering result.

Fig. 6. Precision resulted from running various Algorithms on different datasets. A high value of Precision indicates a better clustering result.

Fig. 7. Entropy resulted from running various Algorithms on different datasets. A low value of Entropy indicates a better clustering result.
In the aim of validating our approach, we will be using the following metrics:

- **Purity**: purity is the percentage of the objects that were classified correctly:
  \[
P_j = \frac{1}{n_j} \max_i (n_i^j)
  \]
  Where \(n_j\) is the size of the cluster \(j\), and \(n_i^j\) is the number of correctly assigned objects. The overall purity the clustering is given by:
  \[
P = \frac{\sum_j n_j P_j}{\sum_j n_j}
  \]

- **Rand Index (RI)**: [20] The Rand index measures the percentage of correct decisions. It is calculated as follows:
  \[
  RI = \frac{TP+TN}{TP+FP+FN+TN}
  \]
  Where TP is a true positive, TN is a true negative, FP is a false positive, and FN is a false negative.

- **Adjusted Rand Index**: [21] is the corrected-for-chance version of the Rand index, it is given as follows:
  \[
  ARI = \frac{\sum_i (n_i) - \left[ \frac{\sum_i (n_i^j) \cdot \sum_i (n_i^{j*})}{\binom{n}{2}} \right]}{\frac{\binom{n}{2}}{2} \left[ \frac{\sum_i (n_i^j) \cdot \sum_i (n_i^{j*})}{\binom{n}{2}} \right] - \left[ \sum_i (n_i^j) \cdot \sum_i (n_i^{j*}) \right] / \binom{n}{2}}
  \]

- **F-measure**: [22] F-Measure provides a single score that balances both the concerns of precision and recall in one number, it is given by the formula:
  \[
  F_w = \frac{(w^2+1)\cdot P\cdot R}{w^2 \cdot P + R},
  \]
  \[
  E_j = \frac{1}{\log R} \sum_{i=1}^{k} \frac{n_j}{n} \log \frac{n_j}{n}
  \]
  Where: \(k\) is the number of clusters in the dataset.
  Where: \(w\) is a positive real value, \(P\) is the precision, and \(R\) is the recall:
  \[
  P = \frac{TP}{TP+FP}
  \]
  \[
  R = \frac{TP}{TP+FN}
  \]

- **Entropy**: [23] The entropy shows how the various classes of objects are distributed within each cluster, it is given by the following formula:
  \[
  Entropy = \sum_{i=1}^{k} \frac{n_i}{n} E_j
  \]
  Where \(E_j\) is the entropy of cluster \(j\), it is given as follows:
  \[
  E_j = \frac{1}{\log R} \sum_{i=1}^{k} \frac{n_j}{n} \log \frac{n_j}{n}
  \]
  Where: \(k\) is the number of clusters in the dataset.

### B. Results and discussion

Clustering results obtained indicate that our algorithm MOCA-SM obtains overall good results for all datasets (see Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, and Fig. 7) noticing that other algorithms give good results for some datasets and less good results in others. The obtained clustering results are presented by dataset as follows:

1. **Spherical_3_4**: for this dataset, our algorithm besides DBSCAN gives perfect clustering, as it is shown in the result figures. While K-means gets noticeably less good results, SOM gets the least good results for this dataset in all evaluation metrics.

2. **Dataset_3_2**: Similar to the previous dataset, our algorithm and DBSCAN performed a perfect clustering, while K-means and SOM produced less good quality clustering; and it is valid for all evaluation metrics.

3. **Spiralsquare**: for this challenging synthetic dataset, all four algorithms performed worse than the first two datasets. Although k-means slightly beats our algorithm, both of them obtained the best clustering results in terms of cluster purity, Rand Index, and entropy while DBSCAN and SOM get less than average results for this dataset.

   SOM gives the best precision result, closely followed by our algorithm. Regarding F-measure and ARI, SOM gave the best results.

4. **Iris**: for this real-world dataset, our algorithm gives the best results in all evaluation metrics, closely followed by k-means, followed by DBSCAN and SOM which get close results for Iris dataset.

5. **Wine**: for this dataset, SOM gets the best results closely followed by our algorithm for all metrics except for the precision where k-means takes the lead. For this dataset, DBSCAN results were not interpreted because the algorithm used eliminated 80% of the dataset points considering them as noise.

A comparison of the four algorithms for each clustering evaluation metric is presented in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, and Fig. 7.

### V. CONCLUSION AND FUTURE WORK

In this work, we have used congestion games with player-specific functions to model and to resolve the clustering problem. Using game theory tools allow a solid mathematical background to the proposed solution, which is considered to be one advantage of our work besides the good clustering results shown for different datasets.

The proposed algorithm operates on three phases where players are identified to play over a set of objects or data points, and each player aims to improve his gains in terms of connectivity and R-square. In each game played, each player (or cluster-head) plays Nash equilibrium. When all players stop playing, the merging phase starts where cluster-heads decide to merge their clusters until having the final clusters.

Much further work is needed, especially in the 3rd phase to give our algorithm the ability of automatically deciding the number of clusters.

Although scalability results have not been treated in this paper, tests made until now made us think that the proposed clustering approach is very promising in this matter. Those perspectives will surely be covered in future work to enhance our approach and make it one of the most efficient clustering approaches.

### ACKNOWLEDGMENT

The authors wish to thank the Insight Center for Data Analytics, UCD, Dublin, for inviting the first author as a visiting researcher while she was working on the development of this clustering approach.
REFERENCES

[1] E. Biernat and M. Lutz, *Data Science: fondamentaux et études des cas*, 2014.

[2] J. Nash, “NON-COOPERATIVE GAMES,” *Ann. Math. Second Ser. Ann. Math.*, vol. 54, no. 2, pp. 286–295, 1951.

[3] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Diane Cerr. Elsevier Inc., 2006.

[4] D. Xu and Y. Tian, “A Comprehensive Survey of Clustering Algorithms,” *Ann. Data Sci.*, vol. 2, no. 2, pp. 165–193, 2015.

[5] C. Aggarwal and C. Reddy, *DATA Clustering Algorithms and Applications*, 2013.

[6] M. Wooldridge, “Does Game Theory Work?,” 2012.

[7] K. Leyton-Brown and Y. Shoham, “Essentials of Game Theory: A Concise Multidisciplinary Introduction,” *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 2, no. 1, pp. 1–88, 2008.

[8] D. Parkes and S. Seuken, “Game Theory I: Simultaneous-Move Games,” 2013, pp. 17–41.

[9] D. Easley and J. Kleinberg, “Games,” in *Networks, crowds, and markets: Reasoning About a Highly Connected World*, Cambridge University Press, 2010, pp. 155–200.

[10] V. Conitzer and T. Sandholm, “New complexity results about Nash equilibria - tech report,” *Games Econ. Behav.*, vol. 63, no. 2, pp. 621–641, 2008.

[11] C. Daskalakis, “The Complexity of Nash Equilibria,” *Proc. 36th Ann. ACM Symp. Theory Comput.*, pp. 1–201, 2008.

[12] A. Fabrikant, C. Papadimitriou, and K. Talwar, “The complexity of pure Nash equilibria,” in *STOC ’04*, 2004, pp. 604–612.

[13] N. Helouli and G. Veisi, “A Multi-Modal Coevolutionary Algorithm for Finding All Nash Equilibria of a Multi-Player Normal Form Game,” *Ubiquitous Inf. Technol. Appl.*, vol. 214, pp. 21–29, 2013.

[14] I. Myletuchi, “Congestion Games with Player-Specific Payoff Functions,” *Games Econ. Behav.*, vol. 13, no. 1, pp. 111–124, 1996.

[15] R. W. Rosenthal, “A class of games possessing pure-strategy Nash equilibria,” *Int. J. Game Theory*, vol. 2, no. 1, pp. 65–67, 1973.

[16] I. Helouli, M. S. Radjef, and M. T. Kechadi, “Automatic multi-objective clustering based on game theory,” *Expert Syst. Appl.*, vol. 67, pp. 32–48, 2017.

[17] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, “On clustering validation techniques,” *J. Intell. Information Systems*, vol. 17, no. 2–3, pp. 107–145, 2001.

[18] S. Sharma, *Applied multivariate techniques*. New York, USA: John Wiley & Sons, Inc., 1996.

[19] C. Goutte, P. Toft, E. Rostrup, F. Å. Nielsen, and L. K. Hansen, “On clustering fMRI time series,” *Neuroimage*, vol. 9, no. 3, pp. 298–310, 1999.

[20] W. Rand, “Objective criteria for the evaluation of clustering methods,” *J. Am. Stat. Assoc.*, vol. 66, no. 336, pp. 846–850, 1971.

[21] L. Hubert, “Comparing partitions,” *J. Classif.*, vol. 2, pp. 193–198, 1980.

[22] C. Rijsbergen, “Information retrieval,” *London: Butter Worths*, 1979.

[23] Y. Zhao and G. Karypis, “Empirical and theoretical comparisons of selected criterion functions for document clustering,” *Mach. Learn.*, vol. 55, pp. 311–331, 2004.