Article

An Improved Similarity-based Clustering Algorithm for Multi-database Mining

Salim Miloudi [1]*, Yulin Wang [1]* and Wenjia Ding [1]*

[1] School of Computer Science, Wuhan University, Wuhan 430072, China; salim.miloudi@whu.edu.cn (S.M.); yulinwang@whu.edu.cn (Y.W.); w.j.ding@whu.edu.cn (D.W.)

* Correspondence: salim.miloudi@whu.edu.cn (S.M.); yulinwang@whu.edu.cn (Y.W.); w.j.ding@whu.edu.cn (W.D.)

Abstract: Clustering algorithms for multi-database mining (MDM) rely on computing $(n^2 - n)/2$ pairwise similarities between n multiple databases to generate and evaluate $m \in [1, (n^2 - n)/2]$ candidate clusterings in order to select the ideal partitioning which optimizes a predefined goodness measure. However, when these pairwise similarities are distributed around the mean value, the clustering algorithm becomes indecisive when choosing what database pairs are considered eligible to be grouped together. Consequently, a trivial result is produced by putting all the n databases in one cluster or by returning n singleton clusters. To tackle the latter problem, we propose a learning algorithm to reduce the fuzziness in the similarity matrix by minimizing a weighted binary entropy loss function via gradient descent and back-propagation. As a result, the learned model will improve the certainty of the clustering algorithm by correctly identifying the optimal database clusters. Additionally, in contrast to gradient-based clustering algorithms which are sensitive to the choice of the learning rate and require more iterations to converge, we propose a learning-rate-free algorithm to assess the candidate clusterings generated on the fly in a fewer upper-bounded iterations. Through a series of experiments on multiple database samples, we show that our algorithm outperforms the existing clustering algorithms for MDM.

Keywords: Coordinate Descent; Graph Clustering; Multi-database Mining; Convex Optimization; Fuzziness Index; Binary Entropy Loss; Similarity Measure.

1. Introduction

Large multi-branch companies need to analyze their multiple databases to discover useful patterns for the decision-making process. To make global decisions for the entire company, the traditional approach suggests to merge and integrate the local branch-databases into a huge repository called a *data warehouse*, and then we can apply data mining algorithms [1] on the accumulated dataset to mine the global patterns useful for all the branches of the company. However, there are some limitations associated with this approach. For instance, the cost of moving the data over the network, integrating and storing potentially heterogeneous databases could be expensive. Moreover, some branches may not accept sharing their raw data due to the underlying privacy issues. More crucially, integrating large amount of irrelevant data can easily disguise some essential patterns hidden in the multiple databases. To tackle the latter problems, it is suggested to keep the transactional data stored locally and only forward the local patterns mined at each branch database to a central site where they will be clustered into disjoint cohesive pattern-base groups for knowledge discovery. In fact, analyzing the local patterns present in each individual cluster of the multiple databases (MDB) enhances the quality of aggregating novel relevant patterns, and also facilitates the parallel maintenance of the obtained database clusters.

Various clustering algorithms and models have been introduced in the literature, namely spectral-based models [2], hierarchical [3], partitioning [4], competitive learning-based models [5–7] and artificial neural networks (ANNs) based clustering [8–10]. Additionally, clustering could be applied in many domains [11,12] including community discovery in social networks [13,14], image segmentation [15,16] and recommendation systems [17–19]. In this article, we focus on exploring similarity-based clustering models for multi-database mining [20–23], due to their stability, simplicity [24] and robustness in partitioning graphs of n multiple databases into k connected components consisting of similar database objects. Nevertheless, the existing clustering quality measures in [20–23] are non-convex objectives suffering from the existence of local optima. Consequently, identifying the optimal clustering may be a difficult task, as it requires evaluating all the candidate clusterings generated at all the local optima in order to find the ideal clustering.

To address the issues associated with clustroid initialization, preselection of a suitable number of clusters and non-convexity of the clustering quality objectives, we proposed in [25,26] an algorithm named GDMDBClustering, which minimizes a quasi-convex loss function quantifying the quality of the multi-database clustering, without a priori assumptions about which number of clusters should be chosen. Therefore, in contrast to the clustering models proposed in [20–23], GDMDBClustering [25] does not require to produce and assess all the possible candidate classifications in order to find the optimal partitioning. Alternatively, each partitioning is assessed on the fly as it is generated and the clustering algorithm terminates just right after attaining the global minimum of the objective function. However, the existing gradient-based clustering algorithms [25,26] are strongly dependent on the choice of the learning rate η, which influences the number of learning cycles required to find the optimal partitioning. In fact, selecting a larger η value may cause global minimum overshooting and setting a smaller η value may necessitate many learning iterations for the algorithm to converge.

In this paper, we improve upon previous work [25,26] and propose a learning-rate-free (i.e., independent of the learning rate η) algorithm requiring fewer upper-bounded iterations (i.e., the maximum number of iterations is at most $(n^2 - n)/2$) to minimize a clustering convex loss function $\mathcal{L}(\theta)$ using coordinate descent (CD) and back-propagation. Precisely, our proposed algorithm minimizes a quadratic hinge-based loss $\mathcal{L}(\theta)$ over the first largest coordinate variable $\theta_{p,q}$ while keeping the rest of the $\binom{n}{2} - 1$ variables fixed. Then, it minimizes $\mathcal{L}(\theta)$ over the second largest coordinate variable while keeping the rest of the $\binom{n}{2} - 1$ variables fixed, and so on until convergence or until cycling through all the $\binom{n}{2}$ coordinate variables. Consequently, our algorithm becomes faster than GDMDBClustering [25] which is dependent on a learning rate and also requires to minimize the cost over a large set of variables at each iteration. This can be a very challenging problem in contrast to minimizing the loss over one single variable at a time while keeping all the other dimensions fixed.

On the other hand, existing clustering algorithms for multi-database mining (MDM) [20–23,25,26] proceed by computing $(n^2 - n)/2$ pairwise similarities $sim(\mathcal{D}_p, \mathcal{D}_q) \in [0, 1]$ between n multiple databases, and then use these values to generate and evaluate $m \in [1, (n^2 - n)/2]$ candidate clusterings in order to select the ideal partitioning optimizing a given goodness measure. However, when $sim(\mathcal{D}_p, \mathcal{D}_q)_{n\times n}$ $(p = 0 \cdots n-2, q = p + 1 \cdots n-1)$ are distributed around the mean value $\mu = 0.5$, the fuzziness index of the similarity matrix increases and the clustering algorithm becomes uncertain when choosing what database pairs are considered similar and hence eligible to be put into the same cluster. Consequently, a trivial result is produced, i.e., putting all the n databases in one cluster or returning n singleton clusters. To tackle the latter problem, we propose a learning algorithm to reduce the fuzziness in the pairwise similarities by minimizing a weighted binary entropy loss function $\mathcal{H}(\cdot)$ via gradient descent and back-propagation. Precisely, the learned model will force the similarity values above 0.5 to go closer to their maximum value (≈ 1), and let those bellow 0.5 go closer to their minimum value (≈ 0) in a way that minimizes $\mathcal{H}(\cdot)$. This will significantly reduce the associated fuzziness and improve the certainty of the clustering algorithm to correctly identify the optimal database clusters.

The remainder of this paper is organized as follows: Section 2 presents an example motivating the importance of clustering for multi-database mining (MDM) and also reviews traditional clustering algorithms for MDM. Section 3 defines the main concepts related to similarity-based clustering and then introduces the proposed approach and its main components. Section 4 presents and analyzes the experimental results. Finally, Section 5 draws conclusions and highlights potential future work.

2. Motivation and Related Work

2.1. Motivating Example

Prior to mining the multiple databases (MDB) of a multi-branch enterprise, it is essential to cluster these MDB into disjoint and cohesive pattern-base groups sharing an important number of local patterns in common. Then, using local pattern analysis and pattern synthesizing techniques [27–30], one can examine the local patterns in each individual cluster to discover novel patterns, including the *exceptional patterns* [31] and the *high-vote patterns* [32], which are extremely useful when it comes to making special targeted decisions regarding each branch cluster of the same enterprise.

Example 1. Consider the six transactional databases $\mathcal{D} = \cup_{p=1}^{6}\{\mathcal{D}_p\}$ shown in Table 1, where each database $\mathcal{D}_p$ records a set of transactions enclosed in parentheses and each transaction contains a set of items separated by commas. Consider a minimum support threshold $\alpha = 0.5$. The local frequent itemsets, denoted by $FIS(\mathcal{D}_p, \alpha)$, and discovered from each database $\mathcal{D}_p$ are shown in Table 2, such that I_k in each tuple $\langle I_k, supp(I_k, \mathcal{D}_p)\rangle$ of $FIS(\mathcal{D}_p, \alpha)$ is the frequent itemset name and $supp(I_k, \mathcal{D}_p)$, named $support$, is the ration of the number of transactions in $\mathcal{D}_p$ containing I_k to the total number of transactions in $\mathcal{D}_p$.

Table 1. Six transactional databases $\mathcal{D}_p$, for $p = 1 \cdots 6$

Transactional Database $(\mathcal{D}_p)$	**Transactions/Rows**
$\mathcal{D}_1$	$(A, C), (A, B, C), (B, C), (A, B, C, D)$
$\mathcal{D}_2$	$(A, B, C), (B, C), (A, B), (A, C), (A, B, D)$
$\mathcal{D}_3$	$(B, C), (A, D), (B, C, D), (A, B, C)$
$\mathcal{D}_4$	$(E, F, H), (F, H), (F, G, H, I, J)$
$\mathcal{D}_5$	$(E, J), (F, H, J), (E, F, H, J), (F, H)$
$\mathcal{D}_6$	$(E, I), (E, F, H), (F, H, I, J), (E, H, J)$

Table 2. The frequent itemsets (FIs) discovered from each transactional database in Table 1 under a threshold $\alpha = 0.5$

Transactional Database $(\mathcal{D}_p)$	**Frequent Itemsets:** $FIS(\mathcal{D}_p, \alpha)$
$\mathcal{D}_1$	$\langle AC, 0.75\rangle, \langle AB, 0.5\rangle, \langle ABC, 0.5\rangle, \langle BC, 0.75\rangle, \langle C, 1.0\rangle, \langle B, 0.75\rangle, \langle A, 0.75\rangle$
$\mathcal{D}_2$	$\langle AB, 0.6\rangle, \langle C, 0.6\rangle, \langle B, 0.8\rangle, \langle A, 0.8\rangle$
$\mathcal{D}_3$	$\langle BC, 0.75\rangle, \langle D, 0.5\rangle, \langle C, 0.75\rangle, \langle B, 0.75\rangle, \langle A, 0.5\rangle$
$\mathcal{D}_4$	$\langle H, 1.0\rangle, \langle F, 1.0\rangle, \langle FH, 1.0\rangle$
$\mathcal{D}_5$	$\langle E, 0.5\rangle, \langle EJ, 0.5\rangle, \langle J, 0.75\rangle, \langle HJ, 0.5\rangle, \langle FHJ, 0.5\rangle, \langle FJ, 0.5\rangle, \langle H, 0.75\rangle, \langle FH, 0.75\rangle, \langle F, 0.75\rangle$
$\mathcal{D}_6$	$\langle I, 0.5\rangle, \langle J, 0.5\rangle, \langle HJ, 0.5\rangle, \langle F, 0.5\rangle, \langle FH, 0.5\rangle, \langle E, 0.75\rangle, \langle EH, 0.5\rangle, \langle H, 0.75\rangle$

Now, the global support of each itemset $I_k \in \cup_{p=1}^{6}\{FIS(\mathcal{D}_p, 0.5)\}$ is calculated via the synthesizing equation [33] defined as follows:

$$supp(I_k, \mathcal{D}) = \frac{\sum_{p=1}^{n} |\mathcal{D}_p| \times supp(I_k, \mathcal{D}_p)}{\sum_{p=1}^{n} |\mathcal{D}_p|} \tag{1}$$

where $n = 6$ is the total number of databases in $\mathcal{D}$ and $|\mathcal{D}_p|$ is the number of transactions in $\mathcal{D}_p$. For instance, we can calculate the global support of the itemset A as follows:

$$\begin{aligned} supp(A, \mathcal{D}) &= \frac{0.75 \times 4 + 0.8 \times 5 + 0.5 \times 4 + 0 \times 3 + 0 \times 4 + 0 \times 4}{4 + 5 + 4 + 3 + 4 + 4} \\ &= 0.375 < \alpha \end{aligned}$$

After computing the global supports of the rest of the itemsets using (1), no single novel pattern has been found, i.e., $\forall\ I_k \in \cup_{p=1}^{6}\{FIS(\mathcal{D}_p, 0.5)\}, supp(I_k, \mathcal{D}) < 0.5$). The reason is that irrelevant patterns were involved in the synthesizing procedure. Now, if we examine the frequent itemsets in Table 2, we observe that some databases share many patterns in common. Precisely, the six databases seem to form two clusters, C_1=$\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3\}$ and C_2=$\{\mathcal{D}_4, \mathcal{D}_5, \mathcal{D}_6\}$, where each cluster of databases tend to share similar frequent itemsets.

Next, let us use the synthesizing equation (1) on the frequent itemsets coming from every single cluster C_i, such that p=$4 \cdots n$=6 for cluster C_2 and p=$1 \cdots n$=3 for cluster C_1. This time new valid frequent itemsets having a support value above the minimum threshold α are discovered in the two clusters. In fact, $FIS(C_2, 0.5)$=$\{\langle F\ H, 0.727\rangle, \langle F, 0.727\rangle, \langle H, 0.818\rangle\}$ and $FIS(C_1, 0.5)$=$\{\langle C, 0.769\rangle, \langle B, 0.769\rangle, \langle A, 0.692\rangle\}$. The obtained patterns show that a percentage of more than 69% of the total transactions in the cluster C_1 include the itemsets C, B and A. Also, more than 72% of the total transactions in the cluster C_2 include $F\ H$, F and H. Moreover, some associations between itemsets could be derived as well, such that the itemset $\langle F\ H, 0.727\rangle \in FIS(C_2, 0.5)$ suggests that on average, if a customer collects the item H at one of the branches in C_2, they are likely to also buy the item F with a $\frac{supp(FH,C_2)}{supp(H,C_2)} = 88.87\%$ confidence.

From the discovered knowledge, decision makers and stakeholders are going to have a clear idea about the branches which exhibit similar purchasing behaviors, and hence take useful decision accordingly. In fact, appropriate business decisions may be taken regarding each group of similar branches in order to predict potential purchasing patterns, increase the customer retention rate and convince customers to purchase more services in the future. Consequently, exploring and examining individual clusters of similar local frequent itemsets is going to help the discovery of new relevant patterns capable of improving the decision-making quality.

2.2. Prior Work

Djenouri *et al.* [34] have adopted a divide and conquer *mono-database mining* approach to accelerate mining global frequent itemsets (FIs) in large transactional databases. Savasere *et al.*, Zhang and Wu [35,36] have proposed similar work where big transactional databases are divided into k disjoint transaction partitions whose sizes are small enough to be read and loaded to the random access memory. Then, the frequent itemsets (FIs) mined from all the k partitions are synthesized into global FIs using an aggregation function such as the one suggested by Ramkumar and Srinivasan [33]. It is worth noting that for mono-database mining applications, we usually have a direct access to the raw data stored in big transactional databases. On the other hand, for multi-database mining (MDM) applications, it is suggested to keep the transactional data stored locally and only forward the local patterns mined at each branch database to a central site where they will be clustered into disjoint cohesive pattern-base groups for knowledge discovery. As a result, the confidential raw data are kept safe, and also the cost associated with transmitting big amount of data over the network is cut off. Hence, in contrast to clustering the transactional data stored in a single data warehouse, our approach consists of clustering the local patterns mined and forwarded from multi-databases without requiring the number of clusters to be set a priori. Our purpose is to identify the group of databases which share similar patterns, such as the *high-vote patterns* [32] and the *exceptional patterns* [31,37,38] that can be used to make specific decisions in regards to their corresponding branches. In the traditional clustering approach [34–36] applied for mono-database mining , we can only mine the global patterns which are supported by the whole multi-branch company.

The existing clustering algorithms for multi-database [20,21,23,39,40] are based on an agglomerative process which generates hierarchical partitionings at different levels of similarity, where each cluster in a given candidate partitioning is included in another cluster of a partitioning produced at the next similarity level. Regardless of the latter observation, each candidate partitioning is produced without taking into account the use of the clusters generated at the previous similarity levels. As a result, the clustering algorithms in [20,21,23,39,40] unnecessarily reconstruct clusters that have been built at the previous similarity levels. The latter limitation inspired Miloudi *et al.* [22] to design a graph-based algorithm which maintains the classes produced at prior similarity levels in order to produce new subsequent classes out of them. Despite the fact that the experiments done in [22] showed promising results against the prior work [20,21,23,39,40], these algorithms are based on non-convex functions to evaluate the quality of the produced candidate clusterings. Consequently, finding the ideal clustering for which a non-convex function is optimum may be a difficult problem to solve in a short time.

To face the latter problem, Miloudi *et al.*[25,26] turned the clustering problem into a quasi-convex optimization problem solvable via gradient descent and back-propagation. Consequently, an early stopping of the clustering process occurs just right after converging to the global minimum. Hence, by avoiding generating and evaluating unnecessary candidate clusterings, we can significantly reduce the CPU execution time. Even thought traditional clustering algorithm such as k-means [4,41] are intuitive, popular and not hard to implement, they remain sensitive to clustroid initialization, preselection of a suitable number of clusters and non-convexity of the clustering quality objective [42]. The Silhouette plot [43] could be used to find an appropriate number of clusters, but this requires executing k-means multiple times with different number of clusters in order to find the ideal partitioning maximizing the Silhouette objective. As a result, the time performance will be influenced in the case of clustering big high-dimensional datasets. Slightly different, Hierarchical-based clustering algorithms [3] build nested hierarchical levels to visualize the relationships between different objects in the form of dendrograms. Then, it is up to the domain expert or to some non-convex metrics to determine at which level the tree diagram should be cut.

Conversely, the optimization problem formulated in [25,26] is quasi-convex. Therefore, convergence to the global optimum is independent of the initial settings. Furthermore, the proposed gradient-based clustering GDMDBClustering [25] does not need to have the number of clusters as a parameter. Alternatively, the number of clusters becomes a parametric function in the main objective. However, GDMDBClustering is based on the choice of a suitable learning rate, i.e., choosing a small learning rate η may increase the number of iterations and slow down learning the optimal weights, whereas a large η may let the algorithm overshoot the global minimum. To overcome the latter imitation, we propose in this paper a learning-rate-free clustering algorithm, named CDClustering, which minimizes a convex objective function quantifying the clustering quality. For this purpose, we use coordinate descent (CD) and back-propagation to search for the optimal clustering of n multiple database in less than $(n^2 - n)/2$ iterations and without using a learning rate. This makes our algorithm faster than the previous gradient-based clustering algorithm [25,26] which remains dependent on a learning-rate defined based on some prior knowledge of the properties of the loss function. On the other hand, due to the fuzziness of the similarity matrix, which increases when the pairwise similarities are distributed around the mean value, the clustering algorithm becomes indecisive when grouping similar databases together. To face this problem, we design a learning algorithm to adjust the pairwise similarities between n multiple databases, in a way which minimizes a binary entropy loss function quantifying the fuzziness associated with the similarity matrix. Thus, the proposed algorithm becomes crisp in discriminating between the different database clusters.

3. Materials and Methods

In this section, we present our fuzziness reduction model applied to the pairwise similarities between n multiple databases and describe in details our coordinate descent-based clustering approach. Some definitions and notions relevant to this work need to be presented first.

3.1. Background and Relevant Concepts

In this subsection, we define the similarity measure between two transaction databases and present the process of generating and evaluating a given candidate clustering. We also define four clustering validity functions used to evaluate the clustering quality.

3.1.1. Similarity Measure

Each transactional database $\mathcal{D}_p$ is encoded as a hash-table $FIS(\mathcal{D}_p,\alpha_p)$=$\{\cup_{k=1}^{m}\langle I_k, supp(I_k,\mathcal{D}_p)\rangle | supp(I_k,\mathcal{D}_p) \geq \alpha_p\}$, where $p = 0 \ldots n-1$, n is the number of transactional databases, m is the number of frequent itemsets in $\mathcal{D}_p$, I_k is the name of the k-th frequent itemset, $supp(I_k,\mathcal{D}_p) \in [0,1]$ is the support of I_k, which is the ratio of the number of rows in $\mathcal{D}_p$ containing I_k to the total number of rows in $\mathcal{D}_p$, and $\alpha_p \in [0,1]$ is the minimum support threshold corresponding to $\mathcal{D}_p$, such that $supp(I_k,\mathcal{D}_p) \geq \alpha_p$. In this paper, FP-Growth [1] algorithm is used to mine the frequent itemsets in each database $\mathcal{D}_p$ as it only requires two passes over the whole database. Our proposed similarity measure is based on maximizing the number of global frequent itemsets (FIs) synthesized from the local FIs in each cluster. Precisely, to measure the similarity between two transactional databases $\mathcal{D}_q$ and $\mathcal{D}_p$, for $p = 0 \cdots n-2, q = p+1 \cdots n-1$, we define the following function:

$$sim(\mathcal{D}_p,\mathcal{D}_q) = \frac{\sum\limits_{I_k \in \{FIS(\mathcal{D}_p,\alpha_p) \cap FIS(\mathcal{D}_q,\alpha_q)\}} \Psi(I_k,\{\mathcal{D}_p,\mathcal{D}_q\})}{|FIS(\mathcal{D}_p,\alpha_p) \cup FIS(\mathcal{D}_q,\alpha_q)|} \tag{2}$$

where

$$\Psi(I_k,\{\mathcal{D}_p,\mathcal{D}_q\}) = \begin{cases} 1 \text{ , if } supp(I_k,\{\mathcal{D}_p,\mathcal{D}_q\}) \geq \alpha_{p,q} \\ 0 \text{ , otherwise} \end{cases}$$

such that

$$supp(I_k,\{\mathcal{D}_p,\mathcal{D}_q\}) = \frac{supp(I_k,\mathcal{D}_p) \times |\mathcal{D}_p| + supp(I_k,\mathcal{D}_q) \times |\mathcal{D}_q|}{|\mathcal{D}_p| + |\mathcal{D}_q|}$$

and

$$\alpha_{p,q} = \frac{\alpha_p \times |\mathcal{D}_p| + \alpha_q \times |\mathcal{D}_q|}{|\mathcal{D}_p| + |\mathcal{D}_q|}$$

We note that the operator $|\cdot|$ is the cardinality of the set passed in as argument. Multiplying α_p by $|\mathcal{D}_p|$ returns the minimum number of transactions in which a frequent itemset I_k should occur in $\mathcal{D}_p$. Therefore, $\alpha_{p,q}$ is the minimum percentage of transactions from the cluster $C_{p,q} = \{\mathcal{D}_p,\mathcal{D}_q\}$ containing the itemset I_k, i.e., $supp(I_k,C_{p,q}) \geq \alpha_{p,q}$. In fact, *sim* (2) takes into account the local minimum support threshold at each database to calculate a new threshold for each cluster.

Example 2. To demonstrate the efficiency of *sim* (2), let us have three transaction databases, $|\mathcal{D}_1|$=200, $|\mathcal{D}_2|$=300 and $|\mathcal{D}_3|$=200 with their corresponding local frequent itemsets: $FIS(\mathcal{D}_1, 0.2)$=$\{\langle C, 0.2\rangle, \langle B, 0.2\rangle, \langle A, 0.2\rangle\}$, $FIS(\mathcal{D}_2, 0.15)$=$\{\langle E, 0.9\rangle, \langle C, 0.2\rangle, \langle B, 0.2\rangle, \langle A, 0.2\rangle\}$ and $FIS(\mathcal{D}_3, 0.25)$=$\{\langle E, 0.9\rangle\}$ mined at different minimum support threshold values $\alpha_1 = 0.2$, $\alpha_2 = 0.15$ and $\alpha_3 = 0.25$, respectively. Now, clustering the 3 databases using the algorithm BestDatabaseClustering[22] equipped with two different similarity measures, *simi* proposed in [20] and our proposed similarity measure *sim* (2), shows the results reported in Table 3. We note that *goodness* [20] is a clustering quality measure, such that higher the value of *goodness* for a given candidate clustering $\mathcal{C}$, better the quality of $\mathcal{C}$.

Table 3. Clustering the three databases $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}_3$ under the similarity measure *simi* [20] against our proposed measure *sim* (2)

Output	Clustering I under *simi* [20]	Clustering II under *sim* (2)
clusters	$\{\mathcal{D}_1\}, \{\mathcal{D}_2, \mathcal{D}_3\}$	$\{\mathcal{D}_1, \mathcal{D}_2\}, \{\mathcal{D}_3\}$
Similarity *intra-cluster*	.6	.75
Distance *inter-cluster*	1.6	1.75
Measure *goodness* [20]	.2	.5

Table 4. Itemsets synthesized from $C_{2,3} = \{\mathcal{D}_2, \mathcal{D}_3\}$ discovered under *simi* [20] against the itemsets synthesized from $C_{1,2} = \{\mathcal{D}_1, \mathcal{D}_2\}$ discovered under *sim* (2)

Synthesized itemsets I_k	$supp(I_k, C_{2,3})$ under *simi* [20]	$supp(I_k, C_{1,2})$ under *sim* (2)
A	$0.12 < \alpha_{2,3} = 0.19$	$0.2 > \alpha_{1,2} = 0.17$
B	$0.12 < \alpha_{2,3} = 0.19$	$0.2 > \alpha_{1,2} = 0.17$
C	$0.12 < \alpha_{2,3} = 0.19$	$0.2 > \alpha_{1,2} = 0.17$
E	$0.9 > \alpha_{2,3} = 0.19$	$0.54 > \alpha_{1,2} = 0.17$

From Table 3, we notice that using our similarity measure *sim* (2), we have obtained a larger intra-cluster similarity, a larger inter-cluster distance and a larger *goodness* [20]. Now, let us synthesize the global frequent itemsets from the clusters containing more than one database, i.e., $C_{2,3} = \{\mathcal{D}_2, \mathcal{D}_3\}$ and $C_{1,2} = \{\mathcal{D}_1, \mathcal{D}_2\}$. The obtained results are shown in Table 4, such that $\alpha_{2,3} = \frac{300\times0.15+200\times0.25}{300+200} = 0.19$ and $\alpha_{1,2} = \frac{200\times0.2+300\times0.15}{200+300} = 0.17$ are the minimum support thresholds corresponding to $C_{2,3}$ and $C_{2,3}$ respectively. As we can see, the similarity measure *simi* [20] captures only high frequency itemsets ($supp \approx 1$), such as E, and neglects low support frequent itemsets (i.e., whose supports are immediately above the minimum threshold α with $supp \in [\alpha, \alpha + \epsilon]$ and ϵ is a very small number), such as A, B and C. This characteristic gives a high similarity value to database pairs sharing only one or very few high frequency itemsets. On the other hand, database pairs sharing many frequent itemsets with a low support will be assigned a lower similarity. However, once the clustering is done, we will be interested in the patterns discovered from each cluster individually, such as the *high-vote patterns* [32] and the *exceptional patterns* [31]. That is why our similarity measure estimates the patterns post-mined from each cluster $C_{p,q} = \{\mathcal{D}_p, \mathcal{D}_q\}$ in order to compute $sim(\mathcal{D}_p, \mathcal{D}_q)$. Since our similarity measure focuses on maximizing the number of frequent itemsets synthesized from each cluster $C_{p,q} \subseteq \mathcal{D}$, only relevant clusters will be assigned a large similarity value.

3.1.2. Clustering Generation and Evaluation

Let $\mathcal{C}(\mathcal{D}, \delta_i)$=$\{C_1, C_2, \ldots, C_k\}$ be a candidate clustering of $\mathcal{D}$=$\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{n-1}\}$ produced at a given level of similarity $\delta_i \in [0,1]$, such that $\cap_{j=1}^{k}\{C_j\} = \emptyset$ and $\cup_{j=1}^{k}\{C_j\} = \mathcal{D}$. From a graph-theoretic perspective, each cluster C_j represents a connected component in a similarity graph $G = (\mathcal{D}, E)$, and an edge $(\mathcal{D}_p, \mathcal{D}_q)$ is added to the list of edges E if and only if $sim(\mathcal{D}_p, \mathcal{D}_q) \geq \delta_i$, where $p = 0 \cdots n-2, q = p+1 \cdots n-1$.

Initially, $G = (\mathcal{D}, E)$ has no edge, i.e., $E = \emptyset$. Then, at a given similarity level $\delta_i \in [0,1]$, edges $(\mathcal{D}_p, \mathcal{D}_q)$ satisfying $sim(\mathcal{D}_p, \mathcal{D}_q) \geq \delta_i$, are added to E. The level of similarity δ_i $(i = 1 \cdots m)$ is chosen from the list of the m unique sorted pairwise similarities $sim(\mathcal{D}_p, \mathcal{D}_q)$ computed between the n transaction databases, such that $\delta_1 > \delta_2 > \cdots \delta_{i-1} > \delta_i > \delta_{i+1} > \cdots > \delta_m$ and $m \leq (n^2 - n)/2$. After adding all the edges $(\mathcal{D}_p, \mathcal{D}_q)$ at δ_i, each graph component C_j $(j = 1 \cdots k)$ will be representing one database cluster in our candidate partitioning $\mathcal{C}(\mathcal{D}, \delta_i)$. One can then use one of the clustering goodness measures shown in Table 5 to assess the quality of $\mathcal{C}(\mathcal{D}, \delta_i)$.

Once we generate and evaluate all the $m \leq (n^2 - n)/2$ candidate clusterings, we report the global optimum (minimum or maximum) of the goodness measure and compare its corresponding clustering with the ground-truth if it is known or with the clustering generated at the maximum point of the Silhouette coefficient when the ground-truth is unknown. In fact, the Silhouette coefficient $SC(\mathcal{D}) \in [-1, 1]$ proposed in [43,44] (See the last row in Table 5) could be used to verify the correctness of the cluster labels assigned to the n transactional databases. Precisely, a value $SC(\mathcal{D}) \approx 1$ suggests that the n transactional databases are highly matched to their own clusters and loosely matched to their neighboring clusters.

We should note that each clustering goodness measure in Table 5 depends on more than two monotonic functions. For instance, the quality measure *goodness* (See the first row in Table 5) proposed in [20] is based on maximizing both the intra-cluster similarity $W(\mathcal{D})$ (which is a non-decreasing function on the interval [0,1]) and the inter-cluster distance $B(\mathcal{D})$ (which is a non-increasing function on the interval [0,1]), while minimizing the number of clusters $f(\mathcal{D})$ (which is a non-increasing function on the interval [0,1]). Consequently, as it was shown via the experiments done in [25,26], most of the time, the graphs of the objectives functions in Table 5 show a non-convex behavior, which makes identifying the ideal partitioning a hard problem to solve without generating and evaluating all the candidate clusterings generated at the local optima.

Table 5. A summary of the clustering quality measures mentioned in this paper.

Clustering quality [reference]	Function (equation)	Optimal value
[20]	$\boldsymbol{goodness(\mathcal{D}) = B(\mathcal{D}) + W(\mathcal{D}) - f(\mathcal{D})}$ $\begin{cases} B(\mathcal{D}) = \sum_{C_t, C_v \in C; t<v} \sum_{\mathcal{D}_p \in C_t, \mathcal{D}_q \in C_v; p<q} (1 - sim(\mathcal{D}_p, \mathcal{D}_q)) \\ W(\mathcal{D}) = \sum_{C_t \in C} \sum_{\mathcal{D}_p, \mathcal{D}_q \in C_t; p<q} sim(\mathcal{D}_p, \mathcal{D}_q) \times \mathbb{1}\{(\mathcal{D}_p, \mathcal{D}_q) \in E\} \\ f(\mathcal{D}) : \text{number of clusters.} \end{cases}$	max $goodness(\mathcal{D})$ (3)
[23]	$\boldsymbol{goodness^2(\mathcal{D}) = \frac{sum\text{-}dist(\mathcal{D})}{(n^2-n)/2} + \frac{coupling(\mathcal{D})}{(n^2-n)/2} + \frac{f(\mathcal{D})-1}{n-1}}$ $\begin{cases} sum\text{-}dist(\mathcal{D}) = \sum_{C_t \in C} \sum_{\mathcal{D}_p, \mathcal{D}_q \in C_t; p<q} (1 - sim(\mathcal{D}_p, \mathcal{D}_q)) \times \mathbb{1}\{(\mathcal{D}_p, \mathcal{D}_q) \in E\} \\ coupling(\mathcal{D}) = \sum_{C_t, C_v \in C; t<v} \sum_{\mathcal{D}_p \in C_t, \mathcal{D}_q \in C_v; p<q} sim(\mathcal{D}_p, \mathcal{D}_q) \end{cases}$	min $goodness^2(\mathcal{D})$ (4)
[21]	$\boldsymbol{goodness^3(\mathcal{D}) = \frac{intra\text{-}sim(\mathcal{D}) + inter\text{-}dist(\mathcal{D})}{f(\mathcal{D})}}$ $\begin{cases} intra\text{-}sim(\mathcal{D}) = \frac{1}{f(\mathcal{D})} \sum_{C_t \in C} \begin{cases} 1, \lvert C_t \rvert = 1 \\ \frac{\sum_{\mathcal{D}_p, \mathcal{D}_q \in C_t} sim(\mathcal{D}_p, \mathcal{D}_q) \times \mathbb{1}\{(\mathcal{D}_p, \mathcal{D}_q) \in E\}}{(\lvert C_t \rvert^2 - \lvert C_t \rvert)/2}, \lvert C_t \rvert > 1 \end{cases} \\ inter\text{-}dist(\mathcal{D}) = \begin{cases} 0, f(\mathcal{D}) = 1 \\ \sum_{C_t, C_v \in C} \frac{2 \times \sum_{\mathcal{D}_p \in C_t, \mathcal{D}_q \in C_v; p<q} (1 - sim(\mathcal{D}_p, \mathcal{D}_q))}{\lvert C_t \rvert \times \lvert C_v \rvert \times (f(\mathcal{D})^2 - f(\mathcal{D}))}, f(\mathcal{D}) > 1 \end{cases} \end{cases}$	max $goodness^3(\mathcal{D})$ (5)
[43,44]	$\boldsymbol{SC(\mathcal{D}) = \frac{1}{n} \sum_{p=0}^{n-1} s(\mathcal{D}_p)}$ $\begin{cases} s(\mathcal{D}_p) = \begin{cases} \frac{b(\mathcal{D}_p) - a(\mathcal{D}_p)}{\max\{a(\mathcal{D}_p), b(\mathcal{D}_p)\}}, \lvert C_p \rvert > 1; \\ 0, if\ \lvert C_p \rvert = 1 \end{cases} \\ \begin{cases} a(\mathcal{D}_p) = \frac{\sum_{\mathcal{D}_p, \mathcal{D}_q \in C_p, p<q} (1 - sim(\mathcal{D}_p, \mathcal{D}_q)) \times \mathbb{1}\{(\mathcal{D}_p, \mathcal{D}_q) \in E\}}{\lvert C_p - 1 \rvert} \\ b(\mathcal{D}_p) = \min_{\mathcal{D}_p \notin C_q} \frac{1}{\lvert C_q \rvert} \sum_{\mathcal{D}_q \in C_q} (1 - sim(\mathcal{D}_p, \mathcal{D}_q)) \end{cases} \end{cases}$	max $SC(\mathcal{D})$ (6)

3.2. *Similarity Matrix Fuzziness Reduction*

In this subsection, we present our fuzziness reduction model applied to the pairwise similarities between n multiple databases. A motivating example is also shown to demonstrate how our model can improve the quality of the multi-database clustering.

Let $z_{p,q} = \theta_{p,q} \times x_{p,q}$ be a weighted similarity, such that $x_{p,q} = sim(\mathcal{D}_p, \mathcal{D}_q)$ is the similarity value between $\mathcal{D}_p$ and $\mathcal{D}_q$ using equation (2) and $\theta_{p,q}$ is the weight value associated with $x_{p,q}$ and $p = 0 \cdots n-2, q = p+1 \cdots n-1$. Let $g : \mathbb{R} \rightarrow]0,1[$ be a continuous piecewise linear activation function and ∂g be its partial derivative defined as follows:

$$\begin{aligned} g(z_{p,q},\epsilon) &= \max(z_{p,q},\epsilon) - \frac{\operatorname{sgn}(z_{p,q}-1+\epsilon)+1}{2}\,(z_{p,q}-1+\epsilon) \\ \frac{\partial g(z_{p,q},\epsilon)}{\partial z_{p,q}} &= \frac{\operatorname{sgn}(z_{p,q}-\epsilon)+1}{2} - \frac{\operatorname{sgn}(z_{p,q}-1+\epsilon)+1}{2} \end{aligned} \tag{7}$$

The graph plots of $g(z_{p,q},\epsilon)$ and $\frac{\partial g(z_{p,q},\epsilon)}{\partial z_{p,q}}$ with respect to $z_{p,q}$ are depicted in Figure 1 (a). The parameter ϵ ensures that each value $z_{p,q}$ is within the range $[\epsilon, 1-\epsilon]$ such that ϵ is a very small number (e.g., $\epsilon = 1e-7$) forcing $g(z_{p,q},\epsilon)$ to be always above 0 and below 1, so that it can be plugged into our log-based loss function defined in (9).

3.2.1. Fuzziness Index

The fuzziness index of the pairwise similarity vector X^T=$[sim(\mathcal{D}_0,\mathcal{D}_1), sim(\mathcal{D}_0,\mathcal{D}_2), \ldots, sim(\mathcal{D}_{n-2},\mathcal{D}_{n-1})]$, also known as the entropy of the fuzzy set X^T [45], and defined from $\mathbb{R}^{\binom{n}{2}}$ to $[0,1]$, is given as follows:

$$\begin{aligned} Fuzziness(X) &= \frac{-2}{n^2-n}\Big(X^T \cdot \log_2(X) + (1-X^T)\cdot \log_2(1-X)\Big) \\ &= \frac{-2}{n^2-n}\sum_{p=0}^{n-2}\sum_{q=p+1}^{n-1}\big(sim(\mathcal{D}_p,\mathcal{D}_q)\log_2(sim(\mathcal{D}_p,\mathcal{D}_q)) + (1-sim(\mathcal{D}_p,\mathcal{D}_q))\log_2(1-sim(\mathcal{D}_p,\mathcal{D}_q))\big) \end{aligned} \tag{8}$$

The smaller the value of $Fuzziness(X)$, the better the clustering performance, and vice-versa. In fact, reducing the fuzziness of the pairwise similarities will lead to a more crisp decision making when it comes to finding the optimal partitioning of the n multiple databases. Particularly, the fuzziness of the similarity matrix increases when the pairwise values are centered around 0.5, resulting in more confusion when we need to decide whether two databases should be in the same cluster or not.

3.2.2. Proposed Model and Algorithm

To reduce the fuzziness associated with the $(n^2-n)/2$ pairwise similarities between the n transaction databases $\mathcal{D}$=$\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{n-1}\}$, we need to make the similarity values that are above the mean value $\mu = 0.5$ go closer to 1, and adjust the similarity values that are below $\mu = 0.5$ to go closer to 0. To do so, we consider the minimization of the sum of the binary entropy loss functions over the $(n^2-n)/2$ weighed similarity values $z_{p,q} = \theta_{p,q} \times x_{p,q}$ as follows:

$$\begin{aligned} \operatorname*{\mathbf{arg\,min}}_{\theta} \mathcal{H}(\theta,\epsilon) &= \operatorname*{\mathbf{arg\,min}}_{\theta} \frac{2}{n^2-n}\sum_{p=0}^{n-2}\sum_{q=p+1}^{n-1}\mathbf{H}(g(z_{p,q},\epsilon)) \\ &= \operatorname*{\mathbf{arg\,min}}_{\theta} \frac{-2}{n^2-n}\sum_{p=0}^{n-2}\sum_{q=p+1}^{n-1}\big(g(z_{p,q},\epsilon)\log_2(g(z_{p,q},\epsilon)) + (1-g(z_{p,q},\epsilon))\log_2(1-g(z_{p,q},\epsilon))\big) \\ &= \operatorname*{\mathbf{arg\,min}}_{\theta} \frac{-2}{n^2-n}\Big(g(\theta^T\odot X^T,\epsilon)\cdot\log_2(g(\theta\odot X,\epsilon)) + (1-g(\theta^T\odot X^T,\epsilon))\cdot\log_2(1-g(\theta\odot X,\epsilon))\Big) \end{aligned} \tag{9}$$

such that n is the number of databases, $\theta^T = [\theta_{0,1}, \theta_{0,2}, \cdots, \theta_{n-2,n-1}]$ represents the model weight vector, $z_{p,q}$ represents the weighted similarity $\theta_{p,q} \times sim(\mathcal{D}_p,\mathcal{D}_q)$ and $g(z_{p,q},\epsilon)$ is the activation function defined in (7). The graph plots of $\mathbf{H}(g(z_{p,q},\epsilon))$ and

$\frac{\partial \mathbf{H}(g(z_{p,q},\epsilon))}{\partial \mathbf{g}(z_{p,q},\epsilon)}$ with respect to $\mathbf{g}(z_{p,q},\epsilon)$ are depicted in Figure 1 (b). Since the fuzziness of the similarity matrix is influenced by the weights associated with the pairwise similarities, the degree to which a pair of databases $(\mathcal{D}_p, \mathcal{D}_q)$ belongs to the same cluster could be changed by adjusting the corresponding weight $\theta_{p,q}$, which is learned by minimizing (9) via gradient descent and back-propagation. The training equations are derived as follows:

$$\theta_{p,q} = \theta_{p,q} - \eta \frac{\partial \boldsymbol{\mathcal{H}}(\theta,\epsilon)}{\partial \theta_{q,q}} \tag{10}$$

where

$$\begin{aligned} \frac{\partial \boldsymbol{\mathcal{H}}(\theta,\epsilon)}{\partial \theta_{p,q}} &= \frac{-2}{n^2-n}\frac{\partial g(z_{p,q},\epsilon)}{\partial z_{p,q}}\frac{\partial z_{p,q}}{\partial \theta_{p,q}}\log_2\left(\frac{g(z_{p,q},\epsilon)}{1-g(z_{p,q},\epsilon)}\right) \\ &= \frac{-2}{n^2-n}\frac{\partial g(z_{p,q},\epsilon)}{\partial z_{p,q}} x_{p,q}\log_2\left(\frac{z_{p,q},\epsilon)}{1-g(z_{p,q},\epsilon)}\right) \end{aligned} \tag{11}$$

Let η_0 and *epochs* be the initial learning rate and the maximum number of learning iterations, respectively. At each epoch i, the current learning rate η decreases as follows:

$$\eta = \eta_0 \times (1 - i/epochs) \tag{12}$$

We note that selecting a large learning rate value may cause global minimum overshooting, whereas choosing a small learning rate may necessitate many iterations for the algorithm to converge. Hence, it is reasonable to let the learning rate decrease over time as the algorithm converges to the global minimum. In Figure 2 and Algorithm 1 we present in details the framework and the algorithm of the proposed fuzziness reduction model. The proposed learning Algorithm 1: SimFuzzinessReduction keeps adjusting the weight vector θ by moving in the opposite direction to the gradient of the loss function $\boldsymbol{\mathcal{H}}(\theta,\epsilon)$ until it reaches the maximum number of iteration *epochs* or until the magnitude of the gradient vector becomes below the minimum value ϵ. After convergence, we can feed the new similarity values $[g(\theta_0 \times sim(\mathcal{D}_0,\mathcal{D}_1),\epsilon), g(\theta_0 \times sim(\mathcal{D}_0,\mathcal{D}_2),\epsilon),\ldots,g(\theta_0 \times sim(\mathcal{D}_{n-2},\mathcal{D}_{n-1}),\epsilon)]$ to any similarity-based clustering algorithm in order to improve the quality of the produced clustering when the latter is trivial or irrelevant.

```
Algorithm 1: SimFuzzinessReduction
  Data: $n$: number of databases,
        $X^T = [sim(\mathcal{D}_0,\mathcal{D}_1), sim(\mathcal{D}_0,\mathcal{D}_2),\ldots,sim(\mathcal{D}_{n-2},\mathcal{D}_{n-1})]$:
  pairwise similarities
  Parameters: $\theta^T = [\theta_{0,1},\theta_{0,2},\ldots,\theta_{n-2,n-1}]$: weight vector
  Hyper-parameters: $epochs$: max number of learning cycles, $\eta_0$: initial
  learning rate
  Result: $[g(\theta_0 \times sim(\mathcal{D}_0,\mathcal{D}_1),\epsilon), g(\theta_0 \times sim(\mathcal{D}_0,\mathcal{D}_2),\epsilon),\ldots,g(\theta_0 \times$
          $sim(\mathcal{D}_{n-2},\mathcal{D}_{n-1}),\epsilon)] = g(\theta \odot X,\epsilon)$ ▷ adjusted pairwise
          similarities generating the minimum fuzziness index
1 begin
2     $\theta^{(0)} \leftarrow \overrightarrow{1} = [1,1,\cdots,1]$ ▷ Initialize the weight parameters at
       iteration 0
3     $i \leftarrow 1$ ▷ Initialize the iteration counter
4     $\epsilon \leftarrow 1e-7$ ▷ Initialize $\epsilon > 0$, a positive small number which prevents
       passing a zero value to $\log_2$ function
5     while $i < epochs \wedge \|\nabla_{\theta^{(i-1)}}\boldsymbol{\mathcal{H}}(\theta^{(i-1)},\epsilon)\| > \epsilon$ do
6         $\eta \leftarrow \eta_0 \times (1-(i-1) \div epochs)$ ▷ Update the learning rate
7         $\theta^{(i)} \leftarrow \theta^{(i-1)} - \eta\nabla_{\theta^{(i-1)}}\boldsymbol{\mathcal{H}}(\theta^{(i-1)},\epsilon)$ ▷ Update the weight vector
8         $i \leftarrow i+1$
9     return $g(\theta \odot X,\epsilon)$
```

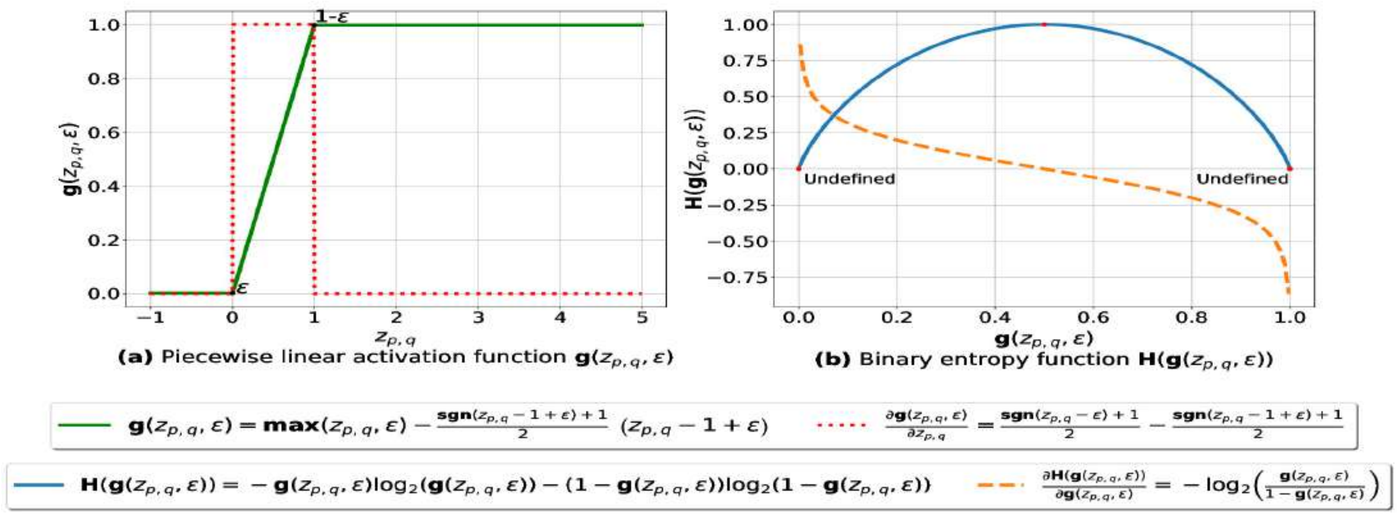


Figure 1. (a): represents (in green) the graph of the piecewise linear activation function $g(\cdot)$ and (in red) its partial derivative. We note that $z_{p,q} = \theta_{p,q} \times x_{p,q}$, and $\theta_{p,q}$ is the weight associated with the similarity value $x_{p,q} = sim(\mathcal{D}_p, \mathcal{D}_q)$, **sgn** : $\mathbb{R} \to \{-1, 1\}$ is the signum function and ϵ is a small number ($\approx 1e - 7$) ensuring that $g(z_{p,q}, \epsilon)$ is always above 0 and below 1. **(b)**: represents the binary entropy function $\mathbf{H} : (0, 1) \to (0, 1]$ in blue and its partial derivative in orange.

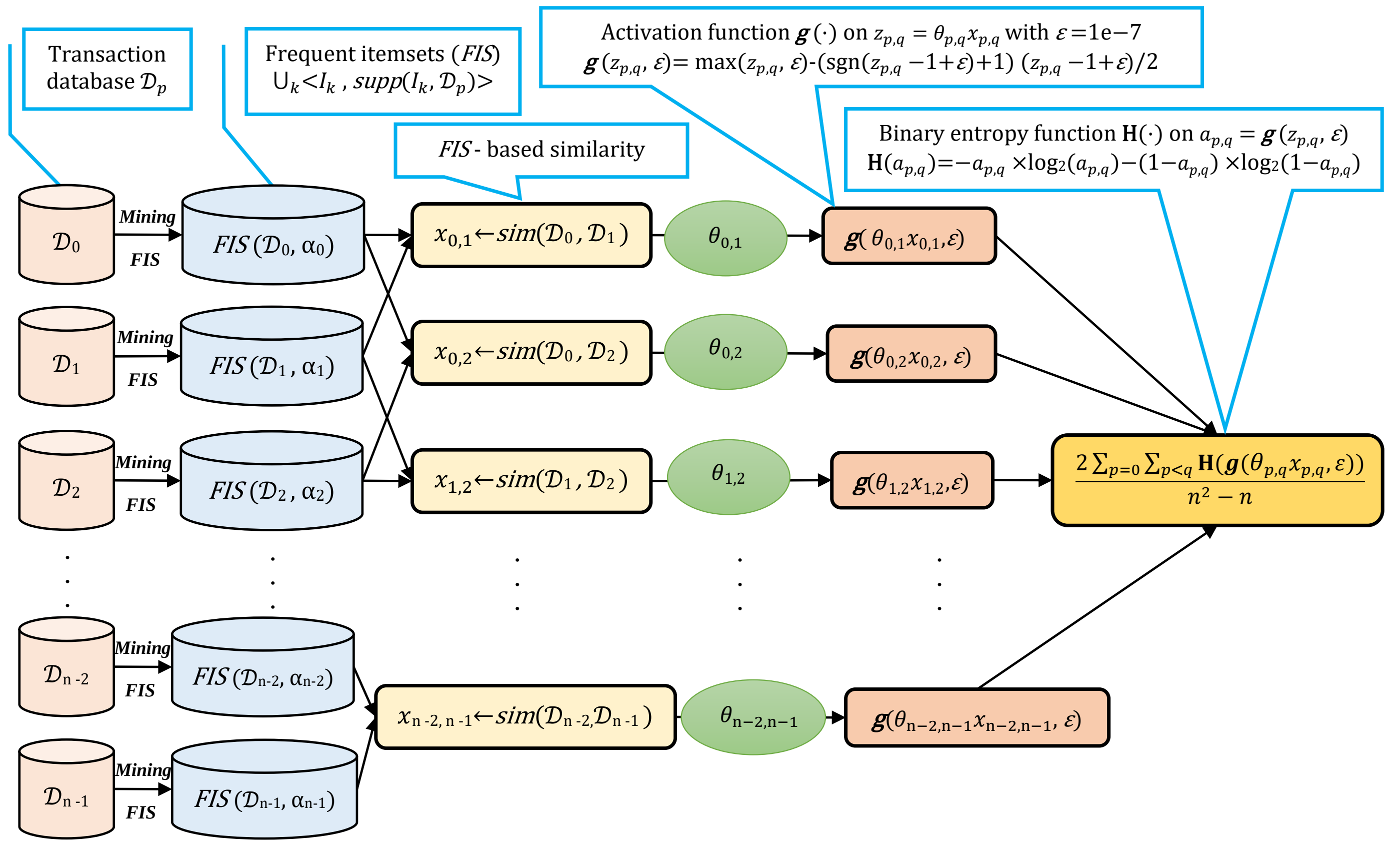

Transaction database $\mathcal{D}_p$
Frequent itemsets (FIS) $\cup_k\langle I_k, supp(I_k, \mathcal{D}_p)\rangle$
FIS - based similarity
Activation function $g(\cdot)$ on $z_{p,q} = \theta_{p,q}x_{p,q}$ with ε=1e−7
$g(z_{p,q}, \varepsilon)$= max$(z_{p,q}, \varepsilon)$-(sgn$(z_{p,q}$ −1+ε)+1) $(z_{p,q}$ −1+ε)/2
Binary entropy function H(·) on $a_{p,q} = g(z_{p,q}, \varepsilon)$
H$(a_{p,q})$=−$a_{p,q}$ ×log2($a_{p,q}$)−(1−$a_{p,q}$) ×log2(1−$a_{p,q}$)
Mining FIS
$x_{0,1} \leftarrow sim(\mathcal{D}_0, \mathcal{D}_1)$
$2\sum_{p=0}\sum_{p<q}\mathbf{H}(g(\theta_{p,q}x_{p,q}, \varepsilon))$ / $n^2 - n$

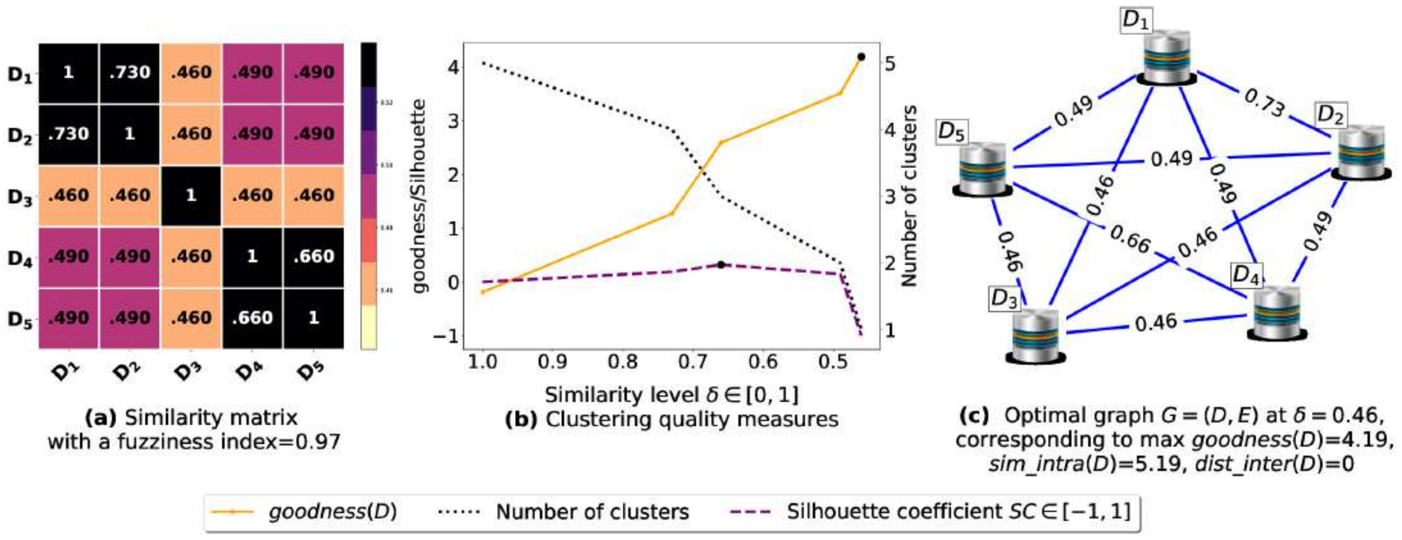
Figure 2. Proposed fuzziness reduction model on the $(n^2 - n)/2$ pairwise similarities $x_{p,q} = sim(\mathcal{D}_p, \mathcal{D}_q)$, $p = 0 \cdots n - 2, q = p + 1 \cdots n - 1$. We note that the graphs corresponding to the activation function $g(\cdot)$ and the binary entropy function $\mathbf{H}(\cdot)$ are plotted in Figure 1.

Example 3. To demonstrate the importance of reducing the fuzziness associated with a similarity matrix, we run the clustering algorithm BestDatabaseClustering [22] on two similarity matrices Figure 3 (a) and Figure 5 (a). The obtained results in terms of the optimal clustering, max $goodness(\mathcal{D})$ [20], the optimal similarity level δ_{opt} (i.e., the similarity level at max $goodness(\mathcal{D})$) and the Silhouette coefficient $SC(\mathcal{D})$ [43] at δ_{opt} are shown in subfigures Figure 3 (b,c) and Figure 5 (b,c) corresponding to the rows 1 and 3 of Table 6, respectively. From the obtained results, we can clearly see that when the similarity matrices are centered around the mean value 0.5, the fuzziness index becomes larger and closer to 1, and BestDatabaseClustering [22] could not return a meaningful clustering since it has put all the n databases into the same cluster.

Now, let us run our fuzziness reduction model on the previous similarity matrices and depict the adjusted similarity matrices in subfigures Figure 4 (a) and Figure 6 (a), respectively. Afterward, we run BestDatabaseClustering [22] on the new similarity matrices and show the clustering results in subfigures Figure 4 (b,c) and Figure 6 (b,c) corresponding to the rows 2 and 4 of Table 6, respectively. As we can see, after reducing the fuzziness index associated with the previous similarity matrices in Figure 3 (a) and Figure 5 (a), the algorithm BestDatabaseClustering [22] was able to produce meaningful non-trivial clusterings with an increase in the Silhouette coefficient $SC(\mathcal{D})$ [43] for both similarity matrices in Figure 4 (a) and Figure 6 (a).

Table 6. A summary of the results obtained in Figures 3-6. We note that δ_{opt} is the optimal similarity level at which $\boldsymbol{goodness(\mathcal{D})}$[20] attains its maximum value, and θ^T is the optimal weight vector learned after a number of *epochs*.

Similarity matrix	Fuzziness index (8)	$\theta^T, epochs, \eta$	max $goodness(\mathcal{D})$[20]	δ_{opt}	$SC(\mathcal{D})$[43,44] at δ_{opt}	Optimal clustering at δ_{opt}
Figure 3	0.97	$\theta^T = [1, 1, \cdots, 1, 1]$ (Without fuzziness reduction)	4.19	0.46	-1	$\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4, \mathcal{D}_5\}$
Figure 4	0.74	θ^T=[1.30,0.52,0.71,0.71,0.52, 0.71,0.71,0.52,0.52,1.44], $epochs = 300$, $\eta = 0.1$	4.54	0.95	0.73	$\{\mathcal{D}_1, \mathcal{D}_2\}, \{\mathcal{D}_3\}, \{\mathcal{D}_4, \mathcal{D}_5\}$
Figure 5	0.95	$\theta^T = [1, 1, \cdots, 1, 1]$ (Without fuzziness reduction)	1.29	0.313	-1	$\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4\}$
Figure 6	0.81	$\theta^T = [0.63, 0.638, 0.591, 0.712, 0.712, 0.77]$, $epochs = 100$, $\eta = 0.1$	1.27	0.292	0.08	$\{\mathcal{D}_4, \mathcal{D}_3, \mathcal{D}_2\}, \{\mathcal{D}_1\}$



(a) Similarity matrix with a fuzziness index=0.97

(b) Clustering quality measures

(c) Optimal graph $G = (D, E)$ at $\delta = 0.46$, corresponding to max $goodness(D)$=4.19, $sim\_intra(D)$=5.19, $dist\_inter(D)$=0

Figure 3. (a): 5×5 similarity matrix example from five transaction databases before applying our fuzziness reduction model. **(b):** represents the graph plots corresponding to $goodness(\mathcal{D})$ [20], the Silhouette coefficient [43] and the number of clusters. **(c):** represents the optimal graph obtained at max $goodness(\mathcal{D})$ [20].

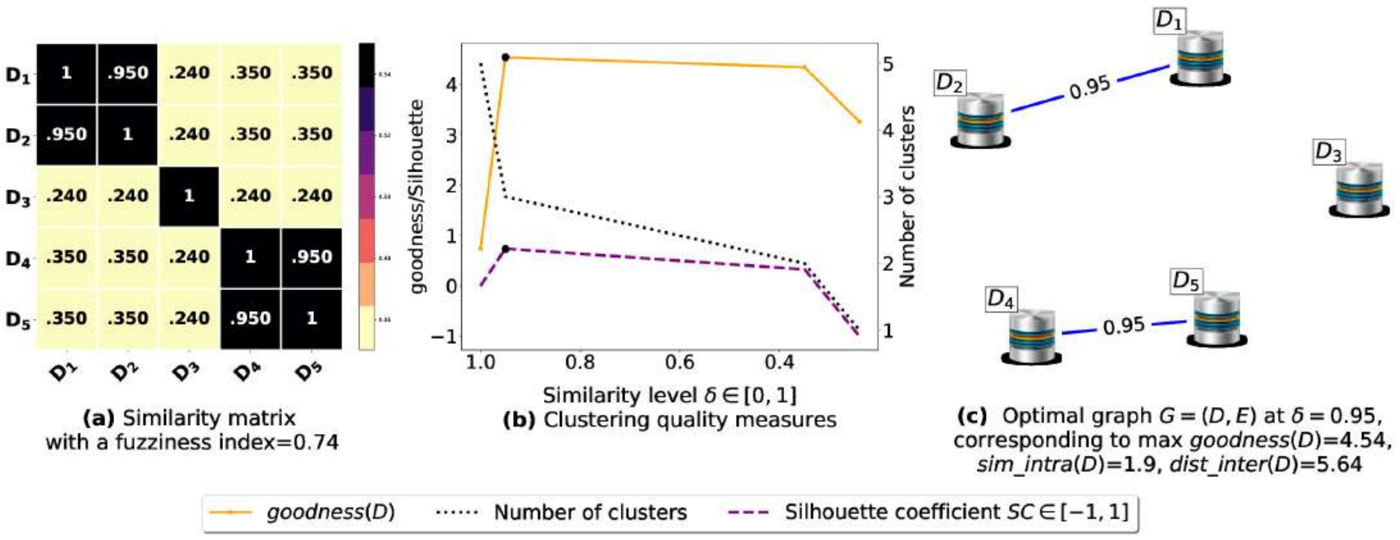


Figure 4. (a): 5×5 similarity matrix obtained after applying our fuzziness reduction model on Figure 3 (a). **(b):** represents the graph plots corresponding to $goodness(\mathcal{D})$ [20], the Silhouette coefficient [43] and the number of clusters. **(c):** represents the optimal graph obtained at max $goodness(\mathcal{D})$ [20].

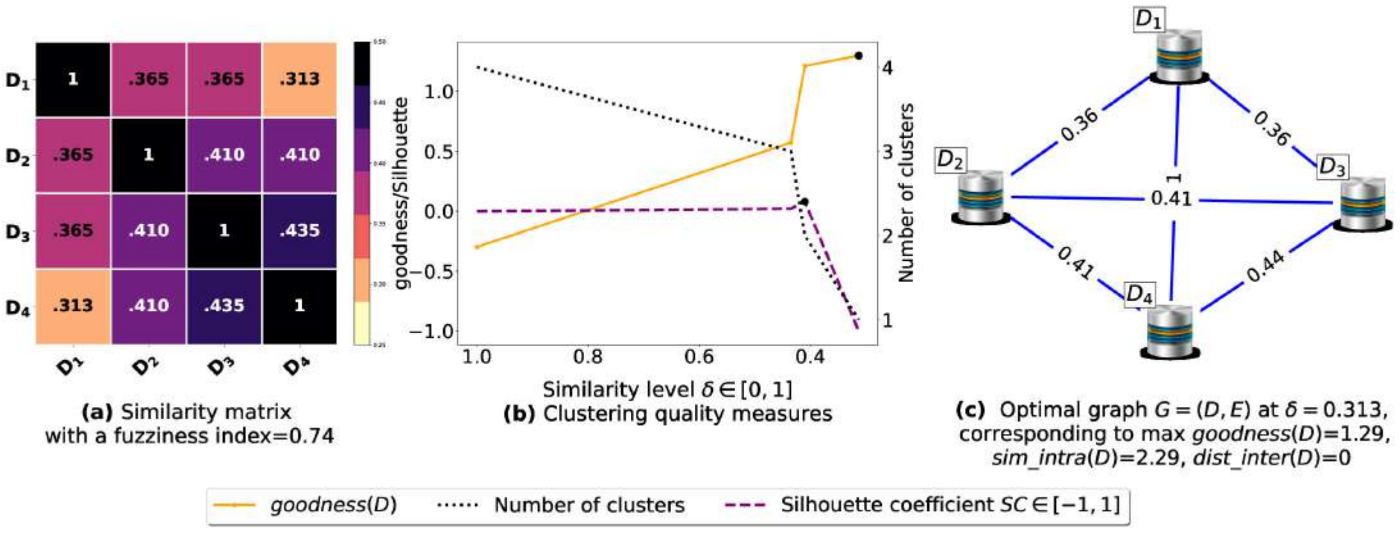


Figure 5. (a): represents a similarity matrix between 4 databases partitioned from Mushroom dataset [46]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p ($p = 1 \cdots 4$) under a threshold $\alpha = 0.5$. **(b):** represents the graph plots corresponding to $goodness(\mathcal{D})$ [20], the Silhouette coefficient [43] and the number of clusters. **(c):** represents the optimal graph obtained at max $goodness(\mathcal{D})$ [20].

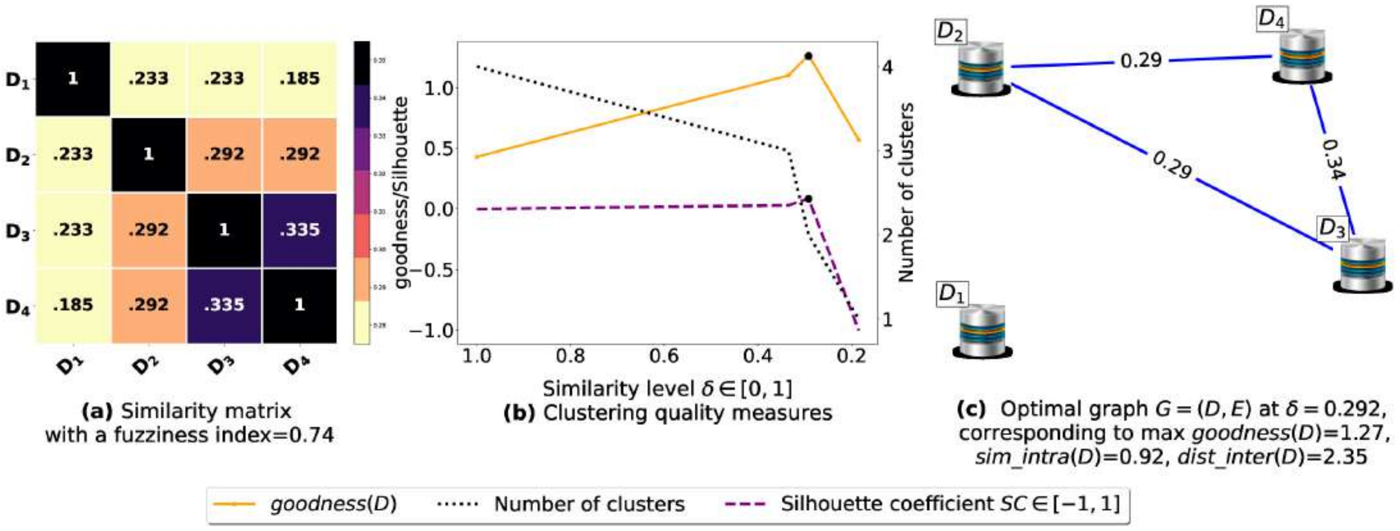


Figure 6. (a): represents the similarity table generated after applying our fuzziness reduction model on Figure 5 (a). **(b):** represents the graph plots corresponding to $goodness(\mathcal{D})$ [20], the Silhouette coefficient [43] and the number of clusters. **(c):** represents the optimal graph obtained at max $goodness(\mathcal{D})$ [20].

3.3. *Proposed Coordinate Descent-based Clustering*

In this subsection, we present and discuss our proposed loss function and our coordinate descent-based clustering approach in details. Unlike the gradient-based clustering in [25,26], our algorithm is learning-rate-free and needs to run at most $(n^2 - n)/2$ learning cycles to converge to the global minimum, such that n is the number of transaction databases. In fact, at each iteration, the largest coordinate variable $\theta_{p,q}$ is selected and popped from a max-heap data structure (initially built by pushing the $(n^2 - n)/2$ pairwise similarities onto the heap). Then, we minimize our quadratic convex hinge-based loss $\mathcal{L}(\theta)$ over $\theta_{p,q}$ which is then adjusted by moving in the opposite direction to the gradient of $\mathcal{L}(\theta)$. This process continues until satisfying a convergence test, which will be defined later in this subsection. Each bloc of selected coordinate variables $\theta_{p,q}$ that have the same value will form a set of edges to be added to our graph $G = (\mathcal{D}, E)$. Determining the disjoint connected components in G after convergence will allow us to discover the optimal database clusters maximizing the intra-cluster similarity and the inter-cluster distance.

3.3.1. Proposed Loss Function and Algorithm

In order to implement our coordinate descent-based clustering, we propose a quadratic version of the hinge loss $\mathcal{L}(\theta) : \mathbb{R}^{\binom{n}{2}} \to [0, \frac{n^2-n}{4}]$, which is a convex function (See Proof of Theorem 1) whose minimization problem is formulated as follows:

$$\underset{\boldsymbol{\theta}^{(i)}}{\textbf{arg min}}\ \mathcal{L}(\theta^{(i)}) = \underset{\boldsymbol{\theta}^{(i)}}{\textbf{arg min}} \sum_{r=0}^{n-2} \sum_{s=r+1}^{n-1} \frac{1}{2} \max(0, 1 - g(\theta_{r,s}^{(i)}))^2 \tag{13}$$

369 A simplified 3D graph plot of $\mathcal{L}(\theta)$ is depicted in Figure 7.

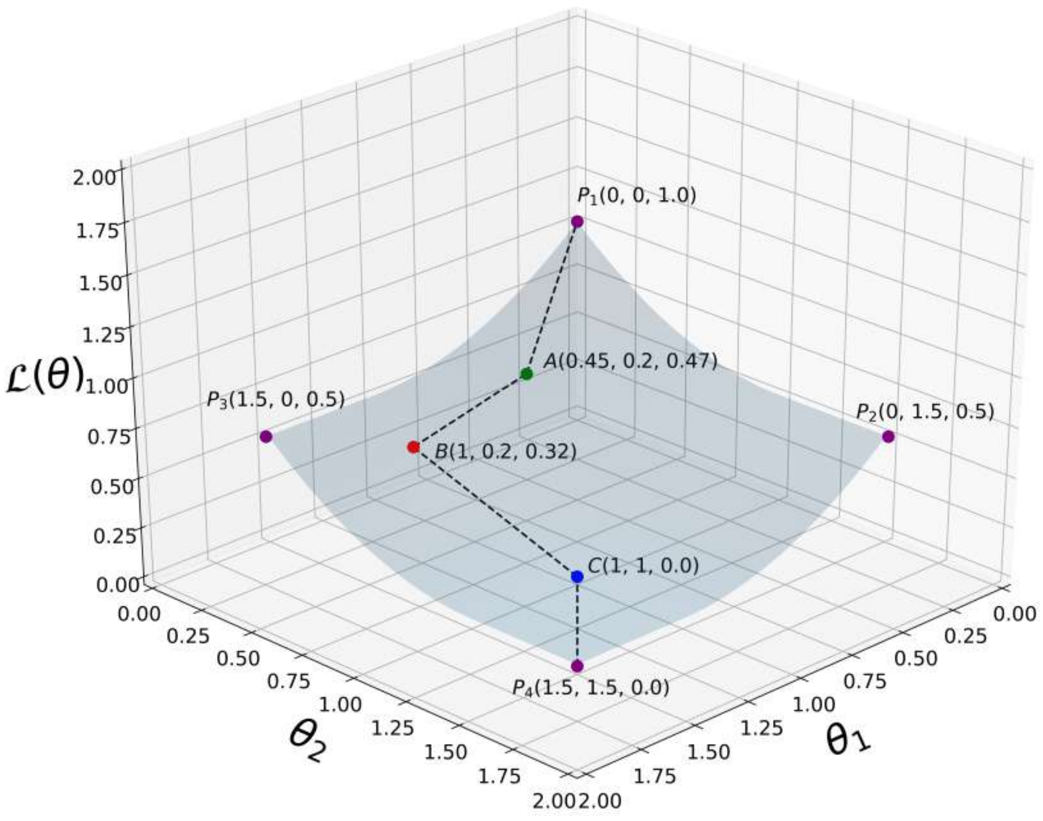


Figure 7. A simplified 3D plot of our proposed loss function $\boldsymbol{\mathcal{L}(\theta)}$ as defined in (13), where $\boldsymbol{\theta} = [\theta_1, \theta_2]$ for visualization purposes. $P_1, P_2, P_3, P_4, A, B, C$ are some selected 3D points at which $\boldsymbol{\mathcal{L}(\theta)}$ is evaluated. From P_1 all the way down to P_4, we can clearly see that $\boldsymbol{\mathcal{L}(\theta)}$ decreases monotonically when the coordinate variables θ_1 and θ_2 increase their values. That is, $\forall(\theta_1^{(i)}, \theta_2^{(i)}, \theta_1^{(i-1)}, \theta_2^{(i-1)}) \in \mathbb{R}^4 | \theta_1^{(i)} \geq \theta_1^{(i-1)} \wedge \theta_2^{(i)} \geq \theta_2^{(i-1)}, \mathcal{L}(\theta_1^{(i)}, \theta_2^{(i)}) \leq \mathcal{L}(\theta_1^{(i-1)}, \theta_2^{(i-1)})$, where i is an integer representing the current iteration in our algorithm.

Initially, the weight vector θ^T is set to the $\binom{n}{2}$ pairwise similarities $X^T = [sim(\mathcal{D}_0, \mathcal{D}_1), sim(\mathcal{D}_0, \mathcal{D}_2), \ldots, sim(\mathcal{D}_{n-2}, \mathcal{D}_{n-1})]$, and then each weight component of θ^T is pushed onto a max-heap data structure. At each iteration $i = 1 \cdots \binom{n}{2}$, the weight $\theta^{(i)}_{p,q}$ ($p = 0 \cdots n-2$, $q = p+1 \cdots n-1$) associated with the current largest similarity value $sim(\mathcal{D}_p, \mathcal{D}_q)$ is popped from the max-heap and is updated as follows:

$$\begin{aligned} \theta^{(i)}_{p,q} &= \theta^{(i-1)}_{p,q} - \eta \frac{\partial \mathcal{L}(\theta^{(i-1)})}{\partial \theta_{q,q}} \\ &= \theta^{(i-1)}_{p,q} + \eta(1 - g(\theta^{(i-1)}_{p,q})) \frac{\partial g(\theta^{(i-1)}_{p,q})}{\partial \theta_{q,q}} \end{aligned} \tag{14}$$

Such that $g : \mathbb{R} \to [0, 1]$ is a differentiable activation function defined as follows:

$$g(\theta_{p,q}) = \max(\theta_{p,q}, 0) - \frac{\mathrm{sgn}(\theta_{p,q} - 1) + 1}{2} \times (\theta_{p,q} - 1) \tag{15}$$

and its partial derivative with respect to the weight $\theta_{p,q}$ is:

$$\frac{\partial g(\theta_{p,q})}{\partial \theta_{q,q}} = \frac{\mathrm{sgn}(\theta_{p,q}) + 1}{2} - \frac{\mathrm{sgn}(\theta_{p,q} - 1) + 1}{2} \tag{16}$$

We note that $\mathrm{sgn} : \mathbb{R} \to \{-1, 1\}$ is the signum function. The usage of $g(\cdot)$ ensures that each weight $\theta_{p,q}$ is within the range $[0, 1]$. As there is no learning rate and schedule to choose for our coordinate descent-based algorithm, we set η to 1.

Theorem 1. $\mathcal{L}(\theta)$ (13) *is convex satisfying the following inequality [47]:*

$$\begin{gathered} \mathcal{L}((1-\varepsilon)\theta^{(i+1)} + \varepsilon\theta^{(i)}) \le (1-\varepsilon)\mathcal{L}(\theta^{(i+1)}) + \varepsilon\mathcal{L}(\theta^{(i)}) \\ \text{for all } \theta^{(i+1)}, \theta^{(i)} \in \mathbb{R}^{\binom{n}{2}} \text{ with } \varepsilon \in [0, 1] \end{gathered} \tag{17}$$

Proof. To prove the convexity of $\mathcal{L}(\theta)$, we can show that its Hessian matrix $\mathbf{H}\ \mathcal{L}$ is positive semi-definite as follows:

$$\mathbf{H}\ \mathcal{L} = \frac{\partial^2 \mathcal{L}}{\partial \theta_{p,q} \partial \theta_{r,s}} = \begin{bmatrix} \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta^2{}_{0,1}} = 1, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{0,1} \partial \theta_{0,2}} = 0, & \cdots, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{0,1} \partial \theta_{n-2,n-1}} = 0 \\ \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{0,2} \partial \theta_{0,1}} = 0, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta^2{}_{0,2}} = 1, & \cdots, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{0,2} \partial \theta_{n-2,n-1}} = 0 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{n-2,n-1} \partial \theta_{0,1}} = 0, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta_{n-2,n-1} \partial \theta_{0,2}} = 0, & \cdots, & \frac{\partial^2 \mathcal{L}(\theta)}{\partial^2 \theta_{n-2,n-1}} = 1 \end{bmatrix}$$

Since $\mathbf{H}$ is positive semi-definite satisfying $\mathbf{x^T H x} \ge 0$ for all $\mathbf{x} \in \mathbb{R}^{\binom{n}{2}}$, $\mathcal{L}(\theta)$ is convex, and therefore guarantees convergence to the global minimum.
□

In order to reach the global minimum of $\mathcal{L}(\theta)$ (i.e., **min** $\mathcal{L}(\theta) = 0$), our learning algorithm needs to set the weight vector $\boldsymbol{\theta}$ to $\overrightarrow{\mathbf{1}}$ (i.e., the unit vector). Consequently, the intra-cluster similarity will reach its maximum value and all the n databases will be put into the same cluster resulting in a meaningless partitioning. Therefore, in order to prevent this scenario from occurring, we need to assess the clustering quality after popping all the coordinate variables that have the same weight $\theta_{p,q}$ (i.e., a block of weights having the same value) from the max-heap. This corresponds to generating one candidate clustering by adding the list of edges $(\mathcal{D}_p, \mathcal{D}_q)$ satisfying $sim(\mathcal{D}_p, \mathcal{D}_q) \ge \theta_{p,q}$ to the graph $G = (\mathcal{D}, E)$. Afterward, we need a stopping condition to terminate our algorithm if the current candidate clustering quality is judged to be the optimal one in terms of the similarity-intra cluster $W_{\theta^{(i)}}(\mathcal{D})$ and the number of clusters $f_{\theta^{(i)}}(\mathcal{D})$. For

doi:10.20944/preprints202104.0256.v1

this purpose, we define the following quasi-convex loss function evaluated at the i- th iteration:

$$\begin{aligned} L(\theta^{(i)}) &= \frac{1}{2}(f_{\theta^{(i)}}(\mathcal{D}) - W_{\theta^{(i)}}(\mathcal{D}))^2 \\ &= \frac{1}{2}(f_{\theta^{(i)}}(\mathcal{D}) - \varphi(\theta^{(i)T}) \cdot \mathrm{X})^2 \\ &= \frac{1}{2}(f_{\theta^{(i)}}(\mathcal{D}) - \sum_{p=0}^{n-2}\sum_{q=p+1}^{n-1} sim(\mathcal{D}_p, \mathcal{D}_q) \times \varphi(\theta^{(i)}_{p,q}))^2 \end{aligned} \tag{18}$$

where $\varphi : \mathbb{R}^{\binom{n}{2}} \rightarrow \{0,1\}^{\binom{n}{2}}$, $\varphi(\theta) = \frac{\mathrm{sgn}(\theta-1)+1}{2}$.

Our algorithm terminates just right after it reaches the global minimum of $L(\cdot)$. In other words, if $L(\theta^{(i)}) \leq L(\theta^{(i-1)})$, then we continue updating the weight vector, the clustering labels, and save the optimal partitioning found so far. Otherwise, the algorithm terminates as it has reached the global minimum $L(\theta^{(i-1)})$, and therefore, the optimal partitioning saved so far is returned as the ideal clustering of the n transactional databases. This stopping condition is only possible due to the quasi-convexity of $L(\cdot)$.

Theorem 2. $L(\theta)$ (18) *is quasi-convex satisfying the following inequality* [47]*:*

$$\begin{gathered} L((1-\varepsilon)\theta^{(i+1)} + \varepsilon\theta^{(i)}) \leq \max\{L(\theta^{(i+1)}), L(\theta^{(i)})\} \\ \text{for all } \theta^{(i+1)}, \theta^{(i)} \in \mathbb{R}^{\binom{n}{2}} \text{ with } \varepsilon \in [0,1] \end{gathered} \tag{19}$$

Proof. To prove the quasi-convexity of $L(\theta)$, we need to demonstrate the validity of (19). First, since $f_\theta(D)$ is a decreasing function on the range [0,1], it is then both quasi-concave and quasi-convex satisfying the following: $f((1-\varepsilon)\theta^{(i+1)} + \varepsilon\theta^{(i)}) \leq \max\{f(\theta^{(i+1)}), f(\theta^{(i)})\}$ for all $\theta^{(i+1)}, \theta^{(i)} \in \mathbb{R}^{\binom{n}{2}}$ with $\varepsilon \in [0,1]$. Also, since $W_\theta(D)$ is an increasing function on the range [0,1], it is then both quasi-concave and quasi-convex satisfying the following: $W((1-\varepsilon)\theta^{(i+1)} + \varepsilon\theta^{(i)}) \geq \min\{W(\theta^{(i+1)}), W(\theta^{(i)})\}$ for all $\theta^{(i+1)}, \theta^{(i)} \in \mathbb{R}^{\binom{n}{2}}$ with $\varepsilon \in [0,1]$. By subtracting the two last inequalities, we obtain: $(f((1-\varepsilon)\theta^{(i+1)} + \varepsilon\theta^{(i)}) - W((1-\varepsilon)\theta^{(i+1)}) + \varepsilon\theta^{(i)}) \leq (\max\{f(\theta^{(i+1)}), f(\theta^{(i)})\} - \min\{W(\theta^{(i+1)}), W(\theta^{(i)})\})$. Since $f(\theta^{(i+1)}) \leq f(\theta^{(i)})$ and $W(\theta^{(i+1)}) \geq W(\theta^{(i)})$, the right side of the resulting inequality is equal to $f(\theta^{(i)}) - W(\theta^{(i)})$, which could be set equal to $\max\{f(\theta^{(i+1)}) - W(\theta^{(i+1)}), f(\theta^{(i)}) - W(\theta^{(i)})\}$. Finally, by squaring and dividing both sides of the inequality by 2, we get a variation on the Jensen inequality for quasi-convex functions [47] as defined in (19). Hence, $L(\theta)$ is quasi-convex. $\square$

3.3.2. Time Complexity Analysis

In this subsection, we analyze the time complexity of our coordinate descent-based clustering algorithm presented in Algorithm 4, named CDClustering, which depends on the two subroutines presented in Algorithm 3: $union$ and Algorithm 2: $cluster$. We note that the superscript i enclosed in round brackets, i.e., $\theta^{(i)}_{p,q}$, is used to indicate the iteration number at which a given variable $\theta_{p,q}$ has been assigned a value. The proposed algorithm takes as argument the $\binom{n}{2}$ pairwise similarities $X^T = [sim(\mathcal{D}_0, \mathcal{D}_1), sim(\mathcal{D}_0, \mathcal{D}_2), \ldots, sim(\mathcal{D}_{n-2}, \mathcal{D}_{n-1})]$ and outputs the optimal clustering minimizing our proposed loss function $\mathcal{L}(\theta)$ (13).

First, the weight vector θ^T is initially set equal to X^T. Afterward, coordinate descent and back-propagation are used to search for the optimal weight vector θ^T which minimizes our hinge-based objective $\mathcal{L}(\theta)$. Through each learning cycle i, one coordinate variable $\theta_{p,q}$ is popped from a max-heap. Then, $\theta_{p,q}$ is updated by making the optimal step in the opposite direction to the gradient of $\mathcal{L}(\theta)$. The weights $\theta_{p,q}$ $(p = 0 \cdots n-2, q = p+1 \cdots n-1)$ attaining the maximum value of 1 will have their corresponding database pairs $(\mathcal{D}_p, \mathcal{D}_p)$ put into the same cluster.

By using a max-heap data structure within our coordinate descent algorithm, we optimally choose the current largest variable $\theta^{(i)}_{p,q}$ at each iteration i such that taking the partial derivative of our loss $\mathcal{L}(\theta)$ with respect to $\theta_{p,q}$ allows us to attain the next steepest descent minimizing $\mathcal{L}(\theta)$ without using a learning rate. This way, the maximum number of iterations required for our algorithm to converge is less than or equal to $(n^2 - n)/2$, i.e., the number of the pairwise similarities.

Initially, the number of clusters $f_\theta(\mathcal{D})$ is set equal to the number of transactional databases n. Then, in order to keep track of the database clusters, their number $f_\theta(\mathcal{D})$ and their sizes, we implement a *disjoint-set* data structure [48], which consists of an array A$[0..n-1]$ of n integers managed by two main operations: *cluster* and *union*. Each cluster C_p is represented by a tree whose root index p satisfies $A[p] = -1$, and a database $\mathcal{D}_q$ belonging to the cluster C_p satisfies $A[q] = p$. Therefore, the *cluster* function is called recursively to find the label assigned to the database index p (passed in as argument) by moving down the tree towards the root (i.e., $A[p] = -1$). On the other hand, the *union* procedure links two disjoint clusters C_p and C_q by making the root of the smaller tree point to the root of the larger one in A$[0..n-1]$. The algorithms corresponding to *union* and *cluster* are presented in Algorithm 3 and Algorithm 2, respectively.

Let $s = \binom{n}{2}$ be the size of the weight vector θ^T. The time complexity of building the max-heap is $O(s)$ and the time complexity of Algorithm 4: CDClustering is $O(s + h\log_2(n))$, such that $h \in [1, s]$ is the number of learning cycles run until global minimum convergence, and $O(\log_2(n))$ is the time complexity of one pop operation from the heap. The proposed model is also illustrated in Figure 8. Since it is meaningless to return a single cluster consisting of all the n databases, if the clustering obtained at step (10a) is trivial (i.e., all the n databases are put together in one class or each single database stands alone in its own cluster), then we first need to run the model proposed in Figure 2 on the pairwise similarities to reduce the associated intrinsic fuzziness measured in (8). Afterward, we can apply the proposed model Figure 8 on the new adjusted similarity values to obtain more relevant results.

Algorithm 2: *cluster*

```
Data:   A[0..n-1]: Array representing the n cluster labels
            p: the index of $\mathcal{D}_p$ in A[0..n-1]
Result: The cluster label of $\mathcal{D}_p$ in A[0..n-1]
begin
    if $A[p] \geq 0$ then
        ▷ If p is not a root node then move down the tree towards the root
        A[p] ← cluster(A[p], A)
        return A[p]
    end
    else
        return p ▷ return the root
    end
end
```

Algorithm 3: *union*

```
Data: A[0..n-1]: Array representing the n cluster labels
          $C_p$, $C_q$: Two cluster labels (root nodes)
Result: Applies an updating operation on array A
begin
    if $|A[C_p]| > |A[C_q]|$ then
        $A[C_p] \leftarrow A[C_p] + A[C_q]$ ▷ sum up the two cluster sizes
        $A[C_q] \leftarrow C_p$ ▷ establish a link to the root of the larger cluster
    end
    else
        $A[C_q] \leftarrow A[C_q] + A[C_p]$
        $A[C_p] \leftarrow C_q$
    end
end
```

Algorithm 4: CDClustering

```
Data: $n$: number of databases,
      $X^T = [sim(\mathcal{D}_0,\mathcal{D}_1), sim(\mathcal{D}_0,\mathcal{D}_2),\ldots, sim(\mathcal{D}_{n-2},\mathcal{D}_{n-1})]$:
      pairwise similarity values
Result: $\mathrm{A}_{best}[0\ldots n-1]$: cluster labels assigned to the $n$ databases,
        $f_{\theta^{(i)}}(\mathcal{D})$: number of clusters.
begin
    $\theta^{(0)} \leftarrow X$ ▷ Initialize the weight parameters at iteration 0
    build a max-heap from the weight components $\theta^{(0)}_{p,q}$, for
     $p = 0\cdots n-2,\ q = p+1\cdots n-1$
    for $p \leftarrow 0$ to $n-1$ do $(\mathrm{A}[p], \mathrm{A}_{best}[p]) \leftarrow (-1,-1)$ ▷ Initialize the
     labels assigned to the $n$ databases
    $f_{\theta^{(0)}}(\mathcal{D}) \leftarrow n$ ▷ Initialize the number of clusters
    $W_{\theta^{(0)}}(\mathcal{D}) \leftarrow X^T \cdot \varphi(\theta^{(0)})$ ▷ Initialize the intra-graph similarity
    $L(\theta^{(0)}) \leftarrow (0.5) \times (f_{\theta^{(0)}}(\mathcal{D}) - W_{\theta^{(0)}}(\mathcal{D}))^2$ ▷ Compute the convergence
     test function at iteration 0
    $i \leftarrow 1$ ▷ Initialize the iteration counter
    while heap is not empty $\wedge f_{\theta^{(i-1)}}(\mathcal{D}) > 1$ do
        ▷ Get the weight variable associated with the current largest
         similarity
        $\theta^{(i-1)}_{p,q} \leftarrow$ pop from heap
        ▷ Update the weight variable at the current $\langle p,q\rangle$ coordinates
        $\theta^{(i)}_{p,q} \leftarrow \theta^{(i-1)}_{p,q} + (1 - g(\theta^{(i-1)}_{p,q}))\dfrac{\partial g(\theta^{(i-1)}_{p,q})}{\partial\theta_{q,q}}$
        $C_p \leftarrow \boldsymbol{cluster}(p, \mathrm{A})$ ▷ Get the cluster label assigned to $\mathcal{D}_p$
        $C_q \leftarrow \boldsymbol{cluster}(q, \mathrm{A})$ ▷ Get the cluster label assigned to $\mathcal{D}_q$
        if $C_p \neq C_q$ then
            $\boldsymbol{union}(C_p, C_q, \mathrm{A})$ ▷ Cluster merging: link the smaller
             cluster to the larger one
            $f_{\theta^{(i)}}(\mathcal{D}) \leftarrow f_{\theta^{(i-1)}}(\mathcal{D}) - 1$ ▷ Decrement the number of clusters
             after a union operation
        if peek max from heap $\neq \theta^{(i-1)}_{p,q}$ then
            $W_{\theta^{(i)}}(\mathcal{D}) \leftarrow X^T \cdot \varphi(\theta^{(i)})$ ▷ Compute the intra-graph similarity
             at iteration $i$
            $L(\theta^{(i)}) \leftarrow (0.5) \times (f_{\theta^{(i)}}(\mathcal{D}) - W_{\theta^{(i)}}(\mathcal{D}))^2$ ▷ Compute the
             convergence test function at iteration $i$
            ▷ Test the criterion of optimality
            if $L(\theta^{(i)}) \leq L(\theta^{(i-1)})$ then
                $\mathrm{A}_{best}[0\ldots n-1] \leftarrow \mathrm{A}[0\ldots n-1]$ ▷ Store the current
                 clustering
            else
                break while loop ▷ Algorithm terminates after
                 statisfying the global optimality criterion
        $i \leftarrow i + 1$
    return$(\mathrm{A}_{best}, f_{\theta^{(i-1)}}(\mathcal{D}))$
```

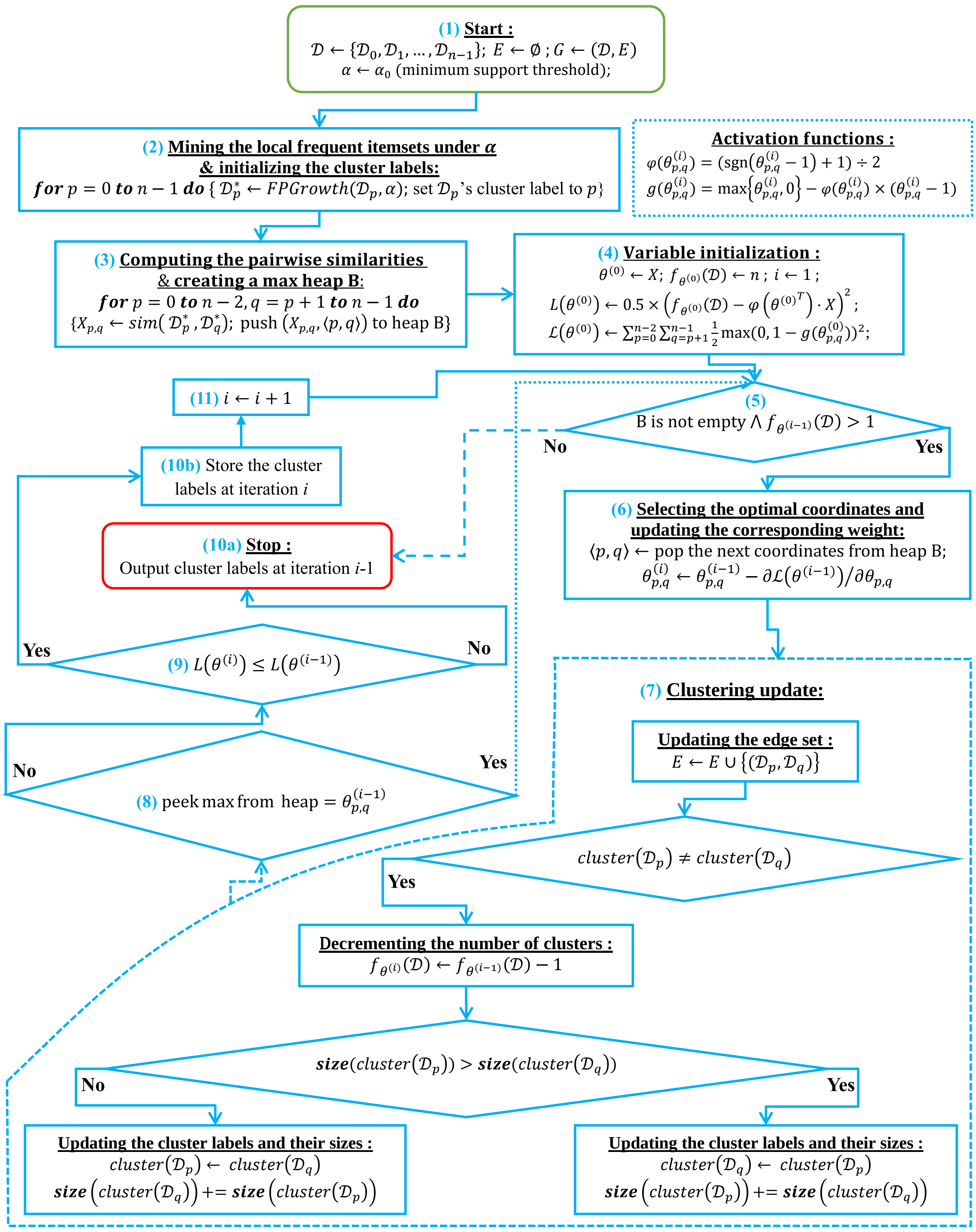

(1) Start :
$\mathcal{D} \leftarrow \{\mathcal{D}_0, \mathcal{D}_1, \dots, \mathcal{D}_{n-1}\}$; $E \leftarrow \emptyset$; $G \leftarrow (\mathcal{D}, E)$
$\alpha \leftarrow \alpha_0$ (minimum support threshold);
(2) Mining the local frequent itemsets under α & initializing the cluster labels:
$\textbf{for } p = 0 \textbf{ to } n-1 \textbf{ do } \{ \mathcal{D}_p^* \leftarrow FPGrowth(\mathcal{D}_p, \alpha)$; set $\mathcal{D}_p$'s cluster label to $p\}$
Activation functions :
$\varphi(\theta_{p,q}^{(i)}) = (\mathrm{sgn}(\theta_{p,q}^{(i)} - 1) + 1) \div 2$
$g(\theta_{p,q}^{(i)}) = \max\{\theta_{p,q}^{(i)}, 0\} - \varphi(\theta_{p,q}^{(i)}) \times (\theta_{p,q}^{(i)} - 1)$
(3) Computing the pairwise similarities & creating a max heap B:
$\textbf{for } p = 0 \textbf{ to } n-2, q = p+1 \textbf{ to } n-1 \textbf{ do}$
$\{X_{p,q} \leftarrow sim(\mathcal{D}_p^*, \mathcal{D}_q^*)$; push $(X_{p,q}, \langle p, q\rangle)$ to heap B$\}$
(4) Variable initialization :
$\theta^{(0)} \leftarrow X$; $f_{\theta^{(0)}}(\mathcal{D}) \leftarrow n$; $i \leftarrow 1$;
$L(\theta^{(0)}) \leftarrow 0.5 \times \left(f_{\theta^{(0)}}(\mathcal{D}) - \varphi\left(\theta^{(0)^T}\right) \cdot X\right)^2$;
$\mathcal{L}(\theta^{(0)}) \leftarrow \sum_{p=0}^{n-2} \sum_{q=p+1}^{n-1} \frac{1}{2} \max(0, 1 - g(\theta_{p,q}^{(0)}))^2$;
(11) $i \leftarrow i + 1$
(5) B is not empty $\wedge f_{\theta^{(i-1)}}(\mathcal{D}) > 1$
No
Yes
(10b) Store the cluster labels at iteration i
(6) Selecting the optimal coordinates and updating the corresponding weight:
$\langle p, q\rangle \leftarrow$ pop the next coordinates from heap B;
$\theta_{p,q}^{(i)} \leftarrow \theta_{p,q}^{(i-1)} - \partial\mathcal{L}(\theta^{(i-1)}) / \partial\theta_{p,q}$
(10a) Stop :
Output cluster labels at iteration i-1
Yes
No
(9) $L(\theta^{(i)}) \leq L(\theta^{(i-1)})$
(7) Clustering update:
Updating the edge set :
$E \leftarrow E \cup \{(\mathcal{D}_p, \mathcal{D}_q)\}$
No
Yes
(8) peek max from heap $= \theta_{p,q}^{(i-1)}$
$cluster(\mathcal{D}_p) \neq cluster(\mathcal{D}_q)$
Yes
Decrementing the number of clusters :
$f_{\theta^{(i)}}(\mathcal{D}) \leftarrow f_{\theta^{(i-1)}}(\mathcal{D}) - 1$
$\textbf{size}(cluster(\mathcal{D}_p)) > \textbf{size}(cluster(\mathcal{D}_q))$
No
Yes
Updating the cluster labels and their sizes :
$cluster(\mathcal{D}_p) \leftarrow cluster(\mathcal{D}_q)$
$\textbf{size}(cluster(\mathcal{D}_q)) \mathrel{+}= \textbf{size}(cluster(\mathcal{D}_p))$
Updating the cluster labels and their sizes :
$cluster(\mathcal{D}_q) \leftarrow cluster(\mathcal{D}_p)$
$\textbf{size}(cluster(\mathcal{D}_p)) \mathrel{+}= \textbf{size}(cluster(\mathcal{D}_q))$

Figure 8. The coordinate descent-based clustering model depicted in eleven steps.

4. Performance Evaluation

To assess the performance of the proposed clustering algorithm, we carried out numerous experiments on real and synthetic datasets, including Zoo [46], Iris [46], Mushroom [46] and T10I4D100K [49]. To simulate a multi-database environment, we have partitioned each dataset horizontally into n partitions $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n$, such that $n \in \{12, 10, 6, 4\}$. Afterward, given a minimum support threshold $\alpha \in \{0.5, 0.2, 0.03\}$, we run FP-Growth [1] on each partition $\mathcal{D}_i$ $(i = 1, \cdots n)$ to discover the local frequent itemsets (FIs) corresponding to each partition. All the details related to the partition sizes and their corresponding FIs are shown in Table 8. We note that the fifth column of Table 8 reports the number of FIs discovered in the entire dataset, whereas the most right column of the same table reports the number of FIs aggregated from the local FIs mined from the partitions in each cluster.

The proposed similarity measure sim (2) is called on the $(n^2 - n)/2$ pairs of FIs to compute the $n \times n$ similarity matrices shown in Figures 9-15 (a). Next, using the obtained pairwise similarities, candidate clusterings are produced via the process described in Subsection 3.1.2, and then evaluated using the clustering quality measures defined in Table 5, including $SC(\mathcal{D})$ [43], $goodness^3(\mathcal{D})$[21], $goodness^2(\mathcal{D})$[23], $goodness(\mathcal{D})$[20] and our proposed loss function $\mathcal{L}(\theta)$ (13). The graphs corresponding to the studied goodness measures are shown in Figures 9-15 (b), where the optimal point (maximum or minimum) of each objective function is depicted as a black dot on its corresponding graph, except that for the graph of our loss function $\mathcal{L}(\theta)$, there is a red dot representing the value $\mathcal{L}(\arg\min_{\theta} L(\theta))$ (i.e., the optimal point at which our algorithm terminates). It is worth mentioning that due to scale differences, we sometimes multiply or divide our loss function $\mathcal{L}(\theta)$, $goodness^3(\mathcal{D})$[21] and $goodness^2(\mathcal{D})$[23] by a scaling number to stretch or shrink their graphs in the direction of the y-axis. The experimental results depicted in Figures 9-15 are summarized in Table 9, such that $\delta \in [0, 1]$ is the ideal similarity threshold for which a goodness measure attains its optimal point. Python version 3.9.2 has been used to implement all the algorithms, and the codes were run on a Ubuntu-20.04 server equipped with an Intel(R) Xeon(R) CPU clocked at 2.30GHz with 50 GB available Disk capacity and 12 GB of available RAM.

4.1. Convexity and Clustering Analysis

In this part of our experiments, we analyze the convex behavior of the proposed clustering quality functions $L(\theta)$ (18) and $\mathcal{L}(\theta)$ (13), and we also examine the non-convexity of the existing goodness measures in [20,21,23,43]. Additionally, we compare the clustering produced by our algorithm and the ones generated at the optimal points of the previous compared goodness measures (i.e., at max $goodness(\mathcal{D})$[20], min $goodness^2(\mathcal{D})$[23] and max $goodness^3(\mathcal{D})$[21]) with the underlying ground-truth cluster labels. When the actual clustering is unknown, we replace it with the partitioning obtained at the maximum value of the Silhouette coefficient [43], that is, at max $SC(\mathcal{D})$. All the graphs corresponding to our loss functions and the compared goodness measures in Table 5 are plotted in Figures 9-15 (b), where the x-axis represents the similarity levels δ at which multiple candidate clusterings are generated and evaluated.

Consider the 7×7 similarity matrix shown in Figure 9 (a). From the graphs plotted in Figure 9 (b) and according to the results shown in the first row of Table 9, we can see that using our loss function $\mathcal{L}(\theta)$ and $goodness(\mathcal{D})$[20], we were able to find the optimal clustering $\{C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}, C_2 = \{\mathcal{D}_4\}, C_3 = \{\mathcal{D}_7, \mathcal{D}_6, \mathcal{D}_5\}\}$ at a similarity level $\delta = 0.44$ where the Silhouette coefficient reaches its maximum value $SC(\mathcal{D}) = 0.46$. On the other hand, $goodness^3(\mathcal{D})$[21] and $goodness^2(\mathcal{D})$[23] did not successfully discover the partitioning maximizing the Silhouette coefficient. Additionally, we observe that the proposed convergence test function $L(\theta)$ has a quasi-convex behavior (See Proof of Theorem 2). This allows us to terminate the clustering process just right after reaching the global minimum. Conversely, the graphs corresponding to $goodness^2(\mathcal{D})$[23] and $goodness(\mathcal{D})$[20] have local optima. Consequently, it is required to explore about $(n^2 -$

$n)/2$ similarity levels in order to generate and evaluate all the candidate clusterings possible.

Now, let us examine the results of some experiments that we have conducted on the synthetic and real-world datasets shown in Table 8. From Figure 15 (b) and Figure 10 (b) (the last and second rows of Table 9), we observe that $goodness^3(\mathcal{D})$[21] and $goodness^2(\mathcal{D})$[23] attain their optimal values when all the partition databases are clustered together in one class. The same phenomenon is observed in Figure 15 (b), Figure 14 (b) and Figure 11 (b) (the last, the sixth and the third rows of Table 9), where both $goodness^2(\mathcal{D})$[23] and $goodness(\mathcal{D})$[20] have put all the databases into one cluster.

In contrast, the proposed loss function $\mathcal{L}(\theta)$ has successfully identified the clustering for which the Silhouette coefficient SC is maximum. Precisely, in Figure 15 (b), which corresponds to the last row of Table 9), $\mathcal{L}(\theta)$ was the only clustering quality measure which has properly identified the ideal 7-class clustering at $\delta = 0.846$.

From the obtained graphs in Figures 9-15, we notice that $goodness^3(\mathcal{D})$[21], $goodness^2(\mathcal{D})$[23] and $goodness(\mathcal{D})$[20] are neither quasi-concave nor quasi-convex on the domain $[0, 1]$. As a result, we observe the existence of local optimum points on their corresponding graphs, which makes the search of the global optimum a difficult problem to solve without exploring all the local solutions.

Conversely, we observe that our loss function $\mathcal{L}(\theta)$ (13) is monotonically decreasing all the time and $\mathcal{L}(\hat{\theta}) = 0$ at $\hat{\theta} = \arg\min_{\theta} \mathcal{L}(\theta) = \overrightarrow{\mathbf{1}}$. This corresponds to the similarity level $\delta = 0$ where all the n databases are put into the same single cluster. To avoid this case from occurring, we used the quasi-convex function $L(\theta)$ (18) as a convergence test function to terminate our algorithm at the point $\mathcal{L}(\arg\min_{\theta} L(\theta))$ corresponding to the red dot on the graph of our loss function $\mathcal{L}(\theta)$. Moreover, it is worth noting that for every two real $\binom{n}{2}$-dimensional vectors $\theta^{(i)}$ and $\theta^{(i+1)}$, where $L(\theta^{(i+1)}) \leq L(\theta^{(i)})$, the line that joins the points $(\theta^{(i+1)}, L(\theta^{(i)}))$ and $(\theta^{(i)}, L(\theta^{(i)}))$ remains above $L(\theta)$, which is observed in Figures 9-15. Therefore, using the proposed loss function $\mathcal{L}(\theta)$ (13) along with $L(\theta)$ (18) guarantees global minimum convergence.

In the fifth and the most right columns of Table 8, we compare the number of frequent itemsets (FIs) mined from all the partitions of a given dataset $\mathcal{D}$ with the FIs mined from each single cluster C_j consisting of similar partitions from the same dataset, where $\cap_{j=1}^{k}\{C_j\} = \emptyset$ and $\cup_{j=1}^{k}\{C_j\} = \mathcal{D}$. We notice that mining all the partitions from datasets Iris [46] and Zoo [46] did not result in discovering any valid frequent itemset. Whereas, mining each individual cluster of partitions from the datasets Iris and Zoo has led to the discovery of new patterns in each cluster C_j.

In Table 10, we report the similarity levels δ_{opt} at which the clustering evaluation measures $goodness(\mathcal{D})$ [20], $goodness^2(\mathcal{D})$ [23], $goodness^3(\mathcal{D})$ [21], the Silhouette Coefficient SC [43] and our proposed loss function $\mathcal{L}(\theta)$ attain their optimal values in Figures 9-15. We note that, the fraction $\frac{|\{\delta_1, \cdots, \delta_{stop}\}|}{|\{\delta_1, \cdots, \delta_m\}|}$ in Table 10 represents the number of similarity levels required to test the convergence and terminate divided by the number of all similarity levels m. We note that opt is the index of the optimal similarity level according to a given clustering quality measure. Since our proposed algorithm is based on a convex loss function, we notice that $stop = opt < m$. On the other hand, as for the compared algorithms which are based on non-convex objectives, we notice that $stop = m$. Therefore, our algorithm requires the least number of similarity levels (opt out of m) in order to converge and terminate, which makes our algorithm faster than the compared algorithms in [22], [23], [21] requiring to generate and evaluate all the m candidate clusterings in order to return the optimal one.

All the previous results confirm that using our loss function $\mathcal{L}(\theta)$ (13) along with $L(\theta)$ (18), we have identified the ideal clustering for which the Silhouette Coefficient SC [43] is maximum and we have also improved the quality of the frequent itemsets (FIs) mined from the multiple databases partitioned from the datasets in Table 8.

4.2. Clustering Error and Running Time Analysis

In this experimental part, we compare the running time of the proposed clustering algorithm with the execution times of two clustering algorithms for multi-database mining (MDM), namely GDMDBClustering [25] and BestDatabaseClustering [22], all run on the same random data samples. We also calculate how the clusterings produced by our algorithm and the compared models are different from the ground-truth clustering. For this purpose, we propose an error function in (20) which measures the difference between two given clusterings $\mathcal{P}$ and $\mathcal{Q}$.

First, we generate n=30 to N=120 isotropic Gaussian blobs using scikit-learn generator [50], such that the number of features for each n blobs is set to $random.randint(2, 10)$, while the number of clusters is set to $\lfloor \frac{n}{2} \rfloor$. In Table 7, we present a brief summary of the random blobs generated via scikit-learn [50].

Table 7. A brief summary of the random blobs generated via scikit-learn [50].

Number of random blobs (n)	**Number of centers** $\lfloor \frac{n}{2} \rfloor$	**Number of attributes** (m)
30	15	random.randint(2, 10)
⋮	⋮	⋮
60	30	random.randint(2, 10)
⋮	⋮	⋮
120	60	random.randint(2, 10)

Afterward, we use the *min-max* scaling[51] to normalize each feature (out of the m features) into the interval [0,1]. Then, for each n blobs, every pair of m-dimensional blobs is passed as arguments to the function *sim* (2) in order to compute the $(n^2 - n)/2$ pairwise similarities between the n blobs. We then run the proposed algorithm, GDMDBClustering [25] (with three different learning rate values) and BestDatabaseClustering [22] on each of the $(n^2 - n)/2$ pairwise similarities ($n = 30 \cdots 120$) and plot their running time graphs in Figures 16-18 (a), and then plot the clustering error graphs in Figures 16-18 (b).

Without loss of generality, assume $\mathcal{Q}$ is the ground-truth clustering (i.e., the actual clusters) of the current n blobs $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_n\}$ generated via scikit-learn [50], and assume $\mathcal{P}$ is the partitioning of $\mathcal{D}$ produced by a given clustering algorithm. To measure how far is $\mathcal{P}$ from $\mathcal{Q}$, we propose the error function $\mathcal{E}_n(\mathcal{P}, \mathcal{Q}) \in [0, 1]$ defined as follows:

$$\mathcal{E}_n(\mathcal{P}, \mathcal{Q}) = \frac{|Pairs_{\mathcal{Q}} \setminus Pairs_{\mathcal{P}}| + |Pairs_{\mathcal{P}} \setminus Pairs_{\mathcal{Q}}|}{|Pairs_{\mathcal{Q}}| + |Pairs_{\mathcal{P}}|} \tag{20}$$

where $|Pairs_{\mathcal{P}}|$ is the number of all the database pairs obtained from every cluster in $\mathcal{P}$ and $|Pairs_{\mathcal{P}} \setminus Pairs_{\mathcal{Q}}|$ is the number of all the database pairs that only exist in $Pairs_{\mathcal{P}}$ and that cannot be found in $Pairs_{\mathcal{Q}}$. We note that $\mathcal{E}_n(\mathcal{P}, \mathcal{Q})$ approaches the maximum value of 1 (i.e., $\mathcal{E}_n(\mathcal{P}, \mathcal{Q}) \approx 1$) when $\mathcal{P}$ and $\mathcal{Q}$ are too different and do not share many database pairs in common (i.e., $|Pairs_{\mathcal{P}} \cap Pairs_{\mathcal{Q}}| \approx 0$). Conversely, $\mathcal{E}_n(\mathcal{P}, \mathcal{Q}) \approx 0$ when the clustering $\mathcal{P}$ and $\mathcal{Q}$ are too similar, i.e., they share the maximum number of pairs $(\mathcal{D}_p, \mathcal{D}_q)$.

We also define the average of the $N - n + 1$ clustering errors, which could also be seen as the mean absolute clustering error:

$$\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = \frac{\sum_n^N \mathcal{E}_n(\mathcal{P}, \mathcal{Q})}{N - n + 1} \tag{21}$$

From the obtained results in Figures 16-18 (a), we observe a rapid increase in the running time of BestDatabaseClustering [22] as the number of generated blobs (n) increases linearly. This is due to the fact that BestDatabaseClustering needs to generate and evaluate approximately $(n^2 - n)/2$ candidate clusterings in order to find the optimal clustering for which the non-convex function $goodness(\mathcal{D})$[20] is maximum. In fact, $goodness(\mathcal{D})$ suffers from the existence of local maxima, which requires exploring all the local candidate solutions in order to find the global maximum. On the other hand, using the proposed convex loss function $\mathcal{L}(\theta)$ and the quasi-convex convergence test function $L(\theta)$ allows us to stop the clustering process at $\mathcal{L}(\arg\min_{\theta} L(\theta))$. Consequently, this avoids generating unnecessary candidate clusterings, and hence reduces the CPU overhead. Also, since our algorithm is independent of the learning rate η, the running time of our algorithm is the same in all Figures 16-18 (a). Whereas, the running time of GDMDBClustering [25] increases for smaller learning rate values (e.g., Figure 18) and decreases when we set larger learning rate values (e.g., Figure 17), but this comes at the cost of having an increased clustering error.

Next, by examining the three clustering error graphs in Figures 16-18 (b), we observe that BestDatabaseClustering [22] has the largest clustering error among the three algorithms with a clustering average error $\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = 0.936$. In fact, on average, BestDatabaseClustering [22] tends to group all the current n blobs ($n = 30 \cdots 120$) in one single cluster. On the other hand, our proposed algorithm and GDMDBClustering [25] produce clusterings that are close to the ground-truth clustering predetermined by the scikit-learn generator [50]. In fact, the average clustering error due to our algorithm is $\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = 0.285$. For GDMDBClustering [25], we get $\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = 0.285$ when the learning rate $\eta = 0.0005$ or $\eta = 0.001$, and the error increases to $\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = 0.29$ when $\eta = 0.002$. The average running times and clustering errors of our algorithm, GDMDBClustering [25] and BestDatabaseClustering [22] are summarized in Table 13.

Our algorithm and GDMDBClustering [25] terminate once we reach the global minimum of the convergence test function $L(\theta)$. Consequently, the running times of our algorithm and GDMDBClustering [25] are most of the time shorter than that of BestDatabaseClustering [22]. Overall, the running time of GDMDBClustering [25] stays relatively steady with respect to n. However, GDMDBClustering depends strongly on the learning step size η and its decay rate. On the other hand, our algorithm is learning-rate-free and needs at most (in the worst case) $(n^2 - n)/2$ iterations to converge. Consequently, our proposed algorithm is faster than both BestDatabaseClustering [22] and GDMDBClustering [25].

To illustrate the statistical significant superiority of the proposed clustering model in terms of running time and clustering accuracy, we have applied the Friedman test [52] (under a significance level $\alpha = 0.05$) on the measurements (execution times and clustering errors depicted in Figures 16-18) obtained by our algorithm, BestDatabaseClustering [22] and GDMDBClustering [25] (with three different values for the learning rate η) considering all the random samples in Table 7.

After conducting the Friedman test [52], we obtained the results shown in Tables 14-16, namely the average running time, the average clustering error $\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})}$ (21), the standard deviation (SD), the variance (Var), the critical value (stat) and its p-value for all the tested clustering algorithms, considering all the 91 random samples generated via scikit-learn [50].

We notice that all the obtained results in Tables 14-16 show a p-value that is below the significance level $\alpha = 0.05$. Consequently, the test suggests to reject the null hypothesis stating that the compared clustering models have a similar performance. In fact, the proposed clustering algorithm significantly outperforms the other compared models, as it has the shortest average running time (6.367 milliseconds) and the lowest average clustering error ($\overline{\mathcal{E}(\mathcal{P}, \mathcal{Q})} = 0.285$) among all the compared models.

4.3. Clustering Comparison and Assessment

In the third part of our experiments, we are interested in using some information retrieval measures to compare the clusterings produced by our algorithm and some other clustering algorithms with the ground-truth data.

Let $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_n\}$ be n transactional databases. Let $\mathcal{P} = \{P_1, P_2, \cdots, P_k\}$ be a k-class clustering of $\mathcal{D}$ produced by any given clustering algorithm, and let $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_l\}$ be the ground-truth clustering of the databases in $\mathcal{D}$, such that $\cup_{i=1}^{k}\{P_i\} = \cup_{i=1}^{l}\{Q_i\} = \mathcal{D}$ and $\cap_{i=1}^{k}\{P_i\} = \cap_{i=1}^{l}\{Q_i\} = \emptyset$. Also, let us define $Pairs_{\mathcal{P}}$ and $Pairs_{\mathcal{Q}}$ as the set of database pairs obtained from each cluster of the same clustering. That is, $Pairs_{\mathcal{P}} = \cup_{P_t \in \mathcal{P}} \cup_{\mathcal{D}_r, \mathcal{D}_s \in P_t; r<s} \{(\mathcal{D}_r, \mathcal{D}_s)\}$ and $Pairs_{\mathcal{Q}} = \cup_{Q_t \in \mathcal{Q}} \cup_{\mathcal{D}_r, \mathcal{D}_s \in Q_t; r<s} \{(\mathcal{D}_r, \mathcal{D}_s)\}$. To compare the two clusterings $\mathcal{P}$ with $\mathcal{Q}$, few methods [53–55] could be used. In this paper, we use a pair counting method [56–59] to calculate some information retrieval measures [60,61], including precision, recall, F-measure (i.e., harmonic mean of recall and precision), Rand index [62] and Jaccard index [63] over pairs of databases being clustered together in $\mathcal{P}$ and/or $\mathcal{Q}$. This will allow us to assess whether the predicted database pairs from $\mathcal{P}$ cluster together in $\mathcal{Q}$, i.e., are the discovered database pairs in $Pairs_{\mathcal{P}}$ correct with respect to the underlying true pairs in $Pairs_{\mathcal{Q}}$ from the ground-truth clustering $\mathcal{Q}$.

In Table 17, we show the categories of database pairs which represent the working set of all pair counting measures cited in Table 18. Precisely, **a:** represents the number of pairs that exist in both clusterings $\mathcal{Q}$ and $\mathcal{P}$, **d:** represents the number of pairs that do not exist in either clustering, **b:** is the number of pairs present only in clustering $\mathcal{Q}$, and **c:** is the number of pairs present only in clustering $\mathcal{P}$. By counting the pairs in each category, we get an indicator for agreement and disagreement of the two clusterings being compared.

Example 4. The following example illustrates how to compute the measures defined in Table 18 for two given clusterings $\mathcal{P} = \{\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3\}, \{\mathcal{D}_4, \mathcal{D}_5, \mathcal{D}_6, \mathcal{D}_7\}\}$ and the ground-truth partitioning $\mathcal{Q} = \{\{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3\}, \{\mathcal{D}_4\}, \{\mathcal{D}_5, \mathcal{D}_6, \mathcal{D}_7\}\}$ of seven transaction databases $\mathcal{D} = \cup_{i=1}^{7}\{\mathcal{D}_i\}$. First, let us calculate the following pairing categories:

$$\begin{aligned}
Pairs_{\mathcal{D}} &= \cup_{r=1}^{6} \cup_{s=r+1}^{7} \{(\mathcal{D}_r, \mathcal{D}_s)\}, \\
Pairs_{\mathcal{Q}} &= \{(\mathcal{D}_6, \mathcal{D}_7), (\mathcal{D}_5, \mathcal{D}_7), (\mathcal{D}_5, \mathcal{D}_6), (\mathcal{D}_2, \mathcal{D}_3), (\mathcal{D}_1, \mathcal{D}_3), (\mathcal{D}_1, \mathcal{D}_2)\}, \\
Pairs_{\mathcal{P}} &= \{(\mathcal{D}_6, \mathcal{D}_7), (\mathcal{D}_5, \mathcal{D}_7), (\mathcal{D}_5, \mathcal{D}_6), (\mathcal{D}_4, \mathcal{D}_7), (\mathcal{D}_4, \mathcal{D}_6), (\mathcal{D}_4, \mathcal{D}_5), (\mathcal{D}_2, \mathcal{D}_3), (\mathcal{D}_1, \mathcal{D}_3), \\
&\quad (\mathcal{D}_1, \mathcal{D}_2)\}.
\end{aligned}$$

Then, $a = |Pairs_{\mathcal{Q}} \cap Pairs_{\mathcal{P}}| = 6$, $b = |Pairs_{\mathcal{Q}} \setminus Pairs_{\mathcal{P}}| = 0$, $c = |Pairs_{\mathcal{P}} \setminus Pairs_{\mathcal{Q}}| = 3$, $d = |Pairs_{\mathcal{D}} \setminus (Pairs_{\mathcal{Q}} \cup Pairs_{\mathcal{P}})| = 12$. Therefore, we get the following measures: $F\text{-}measure = 0.8$, $precision = 0.66$, $recall = 1.0$, $Rand = 0.857$, $Jaccard = 0.66$. We note that higher the values of the evaluation measures given in Table 18, better the matching of the clustering $\mathcal{P}$ to its corresponding ground-truth clustering $\mathcal{Q}$.

In Table 11 and Table 12, we report the F-measure [60,61], precision [60,61], recall [60,61], Rand [62] and Jaccard [63] reached by the clustering algorithms in [22], [23], [21] and our proposed algorithm on the datasets shown in Figures 9-15. From Table 11 and Table 12, we notice that our algorithm achieves the best scores against the compared clustering algorithms , considering all the experiments in Figures 9-15.

Table 8. Description of the datasets used in our experiments with their database partitions and the number frequent itemsets (FIs) mined from each database partition $\mathcal{D}_i$ and the number of FIs mined from each cluster C_j under a threshold α.

Dataset name/ref	Number of rows	Number of rows in partition $\mathcal{D}_i$	Number of $FIS(\mathcal{D}_i, \alpha)$ from partition $\mathcal{D}_i$	Number of $FIS(\mathcal{D}, \alpha)$ from dataset $\mathcal{D}$	Ground truth clustering	Number of $FIS(C_j, \alpha)$ from cluster C_j
Mushroom[46] (2 classes)	8124	$\|\mathcal{D}_1\| = 3916$ (C_1) $\|\mathcal{D}_2\| = 1402$ (C_2) $\|\mathcal{D}_3\| = 1402$ (C_2) $\|\mathcal{D}_4\| = 1404$ (C_2)	$\|FIS(\mathcal{D}_1, 0.5)\| = 375$ $\|FIS(\mathcal{D}_2, 0.5)\| = 2063$ $\|FIS(\mathcal{D}_3, 0.5)\| = 32911$ $\|FIS(\mathcal{D}_4, 0.5)\| = 807$	$\|FIS(\mathcal{D}, 0.5)\| = 151$	$C_1 = \{\mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3, \mathcal{D}_2\}$	$\|FIS(C_1, 0.5)\| = 375$ $\|FIS(C_2, 0.5)\| = 1441$
Zoo [46] (7 classes)	101	$\|\mathcal{D}_1\| = 20$ (C_1) $\|\mathcal{D}_2\| = 21$ (C_1) $\|\mathcal{D}_3\| = 10$ (C_2) $\|\mathcal{D}_4\| = 10$ (C_2) $\|\mathcal{D}_5\| = 5$ (C_3) $\|\mathcal{D}_6\| = 6$ (C_4) $\|\mathcal{D}_7\| = 7$ (C_4) $\|\mathcal{D}_8\| = 2$ (C_5) $\|\mathcal{D}_9\| = 2$ (C_5) $\|\mathcal{D}_{10}\| = 4$ (C_6) $\|\mathcal{D}_{11}\| = 4$ (C_6) $\|\mathcal{D}_{12}\| = 10$ (C_7)	$\|FIS(\mathcal{D}_1, 0.5)\| = 24383$ $\|FIS(\mathcal{D}_2, 0.5\| = 30975$ $\|FIS(\mathcal{D}_3, 0.5)\| = 30719$ $\|FIS(\mathcal{D}_4, 0.5)\| = 32767$ $\|FIS(\mathcal{D}_5, 0.5)\| = 20479$ $\|FIS(\mathcal{D}_6, 0.5\| = 65535$ $\|FIS(\mathcal{D}_7, 0.5)\| = 65535$ $\|FIS(\mathcal{D}_8, 0.5)\| = 114687$ $\|FIS(\mathcal{D}_9, 0.5)\| = 98303$ $\|FIS(\mathcal{D}_{10}, 0.5)\| = 53247$ $\|FIS(\mathcal{D}_{11}, 0.5)\| = 57343$ $\|FIS(\mathcal{D}_{12}, 0.5)\| = 28671$	$\|FIS(\mathcal{D}, 0.5)\| = 0$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_5\}$, $C_4 = \{\mathcal{D}_7, \mathcal{D}_6\}$, $C_5 = \{\mathcal{D}_9, \mathcal{D}_8\}$, $C_6 = \{\mathcal{D}_{11}, \mathcal{D}_{10}\}$, $C_7 = \{\mathcal{D}_{12}\}$,	$\|FIS(C_1, 0.5)\| = 25087$ $\|FIS(C_2, 0.5)\| = 28671$ $\|FIS(C_3, 0.5)\| = 2479$ $\|FIS(C_4, 0.5)\| = 49151$ $\|FIS(C_5, 0.5)\| = 57343$ $\|FIS(C_6, 0.5)\| = 45055$ $\|FIS(C_7, 0.5)\| = 28671$
Iris [46] (3 classes)	150	$\|\mathcal{D}_1\| = 25$ (C_1) $\|\mathcal{D}_2\| = 25$ (C_1) $\|\mathcal{D}_3\| = 25$ (C_2) $\|\mathcal{D}_4\| = 25$ (C_2) $\|\mathcal{D}_5\| = 25$ (C_3) $\|\mathcal{D}_6\| = 25$ (C_3)	$\|FIS(\mathcal{D}_1, 0.2)\| = 5$ $\|FIS(\mathcal{D}_2, 0.2)\| = 6$ $\|FIS(\mathcal{D}_3, 0.2)\| = 2$ $\|FIS(\mathcal{D}_4, 0.2)\| = 2$ $\|FIS(\mathcal{D}_5, 0.2)\| = 2$ $\|FIS(\mathcal{D}_6, 0.2)\| = 5$	$\|FIS(\mathcal{D}, 0.2)\| = 0$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_6, \mathcal{D}_5\}$	$\|FIS(C_1, 0.2)\| = 3$ $\|FIS(C_2, 0.2)\| = 1$ $\|FIS(C_3, 0.2)\| = 2$
T10I4D100K [49] (unknown classes)	100,000	$\|D_i\| = 10,000$ rows, $i = 1 \ldots 10$	$\|FIS(\mathcal{D}_1, 0.03)\| = 58$ $\|FIS(\mathcal{D}_2, 0.03)\| = 58$ $\|FIS(\mathcal{D}_3, 0.03)\| = 62$ $\|FIS(\mathcal{D}_4, 0.03)\| = 57$ $\|FIS(\mathcal{D}_5, 0.03)\| = 62$ $\|FIS(\mathcal{D}_6, 0.03)\| = 63$ $\|FIS(\mathcal{D}_7, 0.03)\| = 63$ $\|FIS(\mathcal{D}_8, 0.03)\| = 59$ $\|FIS(\mathcal{D}_9, 0.03)\| = 61$ $\|FIS(\mathcal{D}_{10}, 0.03)\| = 62$	$\|FIS(\mathcal{D}, 0.03)\| = 50$	Seven clusters found via the Silhouette Coefficient [43] $C_1 = \{\mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_2\}$, $C_3 = \{\mathcal{D}_3\}$, $C_4 = \{\mathcal{D}_5, \mathcal{D}_4\}$, $C_5 = \{\mathcal{D}_6\}$, $C_6 = \{\mathcal{D}_7\}$, $C_7 = \{\mathcal{D}_{10}, \mathcal{D}_9, \mathcal{D}_8\}$	$\|FIS(C_1, 0.03)\| = 58$ $\|FIS(C_2, 0.03)\| = 58$ $\|FIS(C_3, 0.03)\| = 62$ $\|FIS(C_4, 0.03)\| = 59$ $\|FIS(C_5, 0.03)\| = 63$ $\|FIS(C_6, 0.03)\| = 59$ $\|FIS(C_7, 0.03)\| = 61$
Figure 9 [20] (unknown classes)	24	$\|\mathcal{D}_1\| = 3$ $\|\mathcal{D}_2\| = 3$ $\|\mathcal{D}_3\| = 3$ $\|\mathcal{D}_4\| = 4$ $\|\mathcal{D}_5\| = 4$ $\|\mathcal{D}_6\| = 3$ $\|\mathcal{D}_7\| = 4$	$\|FIS(\mathcal{D}_1, 0.42)\| = 3$ $\|FIS(\mathcal{D}_2, 0.42)\| = 3$ $\|FIS(\mathcal{D}_3, 0.42)\| = 5$ $\|FIS(\mathcal{D}_4, 0.42)\| = 7$ $\|FIS(\mathcal{D}_5, 0.42)\| = 7$ $\|FIS(\mathcal{D}_6, 0.42)\| = 5$ $\|FIS(\mathcal{D}_7, 0.42)\| = 3$	$\|FIS(\mathcal{D}, 0.42)\| = 0$	Three clusters found via the Silhouette Coefficient [43] $C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$ $C_2 = \{\mathcal{D}_4\}$ $C_3 = \{\mathcal{D}_7, \mathcal{D}_6, \mathcal{D}_5\}$	$\|FIS(C_1, 0.42)\| = 3$ $\|FIS(C_2, 0.42)\| = 7$ $\|FIS(C_3, 0.42)\| = 3$

Table 9. Clustering results illustrated in Figures 9-15 after using the clustering quality measures $\boldsymbol{goodness^3(\mathcal{D})}$ [21], $\boldsymbol{goodness^2(\mathcal{D})}$ [23], the Silhouette Coefficient $\boldsymbol{SC(\mathcal{D})}$ [43], $\boldsymbol{goodness(\mathcal{D})}$ [20] and our proposed objective function $\boldsymbol{\mathcal{L}(\theta)}$, where δ is the level of similarity at which each clustering evaluation/loss function reaches its optimal value.

Dataset	**Silhouette Coefficient**	**Clustering Result Using Proposed Objective**		**Clustering Result/ Optimal Value**		**Clustering Result/ Optimal Value**		**Clustering Result/ Optimal Value**	
name/ref	*max* $SC(\mathcal{D})$	*clusters*	$\mathcal{L}(\arg\min_\theta L(\theta))$	clusters	*max* $goodness(\mathcal{D})$	clusters	*min* $goodness^2(\mathcal{D})$	clusters	*max* $goodness^3(\mathcal{D})$
Figure 9 7×7 [20]	0.46 at $\delta = 0.444$	$C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4\}$, $C_3 = \{\mathcal{D}_7, \mathcal{D}_6, \mathcal{D}_5\}$	2.004 at $\delta = 0.444$	$C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4\}$, $C_3 = \{\mathcal{D}_7, \mathcal{D}_6, \mathcal{D}_5\}$	15.407 at $\delta = 0.444$	$C_1 = \{\mathcal{D}_7, \ldots, \mathcal{D}_1\}$	0.259 at $\delta = 0.065$	$C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_7, \ldots, \mathcal{D}_4\}$	0.728 at $\delta = 0.086$
Figure 10 12×12 *Zoo* [46]	0.41 at $\delta = 0.559$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_5\}$, $C_4 = \{\mathcal{D}_7, \mathcal{D}_6\}$, $C_5 = \{\mathcal{D}_9, \mathcal{D}_8\}$, $C_6 = \{\mathcal{D}_{11}, \mathcal{D}_{10}\}$, $C_7 = \{\mathcal{D}_{12}\}$	7.71 at $\delta = 0.559$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_5\}$, $C_4 = \{\mathcal{D}_7, \mathcal{D}_6\}$, $C_5 = \{\mathcal{D}_9, \mathcal{D}_8\}$, $C_6 = \{\mathcal{D}_{11}, \mathcal{D}_{10}\}$, $C_7 = \{\mathcal{D}_{12}\}$	32.98 at $\delta = 0.559$	$C_1 = \{\mathcal{D}_{12}, \ldots, \mathcal{D}_1\}$	0.57 at $\delta = 0.348$	$C_1 = \{\mathcal{D}_{12}, \ldots, \mathcal{D}_1\}$	0.42 at $\delta = 0.348$
Figure 11 4×4 *Mushroom*[46]	0.08 at $\delta = 0.41$	$C_1 = \{\mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3, \mathcal{D}_2\}$	0.43 at $\delta = 0.41$	$C_1 = \{\mathcal{D}_4, \ldots, \mathcal{D}_1\}$	1.672 at $\delta = 0.365$	$C_1 = \{\mathcal{D}_4, \ldots, \mathcal{D}_1\}$	0.55 at $\delta = 0.365$	$C_1 = \{\mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3, \mathcal{D}_2\}$	0.68 at $\delta = 0.41$
Figure 12 6×6 *Iris*[46]	0.304 at $\delta = 0.3$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_6, \mathcal{D}_5\}$	1.10 at $\delta = 0.3$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_6, \mathcal{D}_5\}$	9.64 at $\delta = 0.3$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_6, \mathcal{D}_5\}$	0.55 at $\delta = 0.3$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4, \mathcal{D}_3\}$, $C_3 = \{\mathcal{D}_6, \mathcal{D}_5\}$	0.44 at $\delta = 0.3$
Figure 13 6×6 *Zoo* & *Mushroom*[46]	0.63 at $\delta = 0.384$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_6, \ldots, \mathcal{D}_3\}$	0.5 at $\delta = 0.384$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_6, \ldots, \mathcal{D}_3\}$	9.96 at $\delta = 0.384$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_6, \ldots, \mathcal{D}_3\}$	0.40 at $\delta = 0.384$	$C_1 = \{\mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_6, \ldots, \mathcal{D}_3\}$	0.85 at $\delta = 0.384$
Figure 14 4×4 [39]	0.34 at $\delta = 0.429$	$C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4\}$	1.12 at $\delta = 0.429$	$C_1 = \{\mathcal{D}_4, \ldots, \mathcal{D}_1\}$	2.708 at $\delta = 0.25$	$C_1 = \{\mathcal{D}_4, \ldots, \mathcal{D}_1\}$	0.38 at $\delta = 0.25$	$C_1 = \{\mathcal{D}_3, \mathcal{D}_2, \mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_4\}$	0.81 at $\delta = 0.429$
Figure 15 10×10 T10I4D100K [49]	0.115 at $\delta = 0.846$	$C_1 = \{\mathcal{D}_1\}$, $C_2 = \{\mathcal{D}_2\}$, $C_3 = \{\mathcal{D}_3\}$, $C_4 = \{\mathcal{D}_5, \mathcal{D}_4\}$, $C_5 = \{\mathcal{D}_6\}$, $C_6 = \{\mathcal{D}_7\}$, $C_7 = \{\mathcal{D}_{10}, \mathcal{D}_9, \mathcal{D}_8\}$	0.71 at $\delta = 0.846$	$C_1 = \{\mathcal{D}_{10}, \ldots, \mathcal{D}_1\}$	35.275 at $\delta = 0.737$	$C_1 = \{\mathcal{D}_{10}, \ldots, \mathcal{D}_1\}$	0.193 at $\delta = 0.737$	$C_1 = \{\mathcal{D}_{10}, \ldots, \mathcal{D}_1\}$	0.806 at $\delta = 0.737$

Table 10. The similarity levels δ_{opt} at which the clustering evaluation measures $\boldsymbol{goodness^3(\mathcal{D})}$ [21], $\boldsymbol{goodness^2(\mathcal{D})}$ [23], the Silhouette Coefficient $\boldsymbol{SC(\mathcal{D})}$ [43], $\boldsymbol{goodness(\mathcal{D})}$ [20] and our proposed objective function $\boldsymbol{\mathcal{L}(\theta)}$ attain their optimal values in Figures 9-15. The fraction $\frac{|\{\delta_1,\cdots,\delta_{stop}\}|}{|\{\delta_1,\cdots,\delta_m\}|}$ represents the number of similarity levels required to test the convergence and terminate divided by the number of total similarity levels m.

Dataset	**Silhouette Coefficient** $\max_\theta SC(\mathcal{D})$		**Proposed loss function** $\mathcal{L}(\arg\min_\theta L(\theta))$		**Goodness measure** $\max_\theta goodness(\mathcal{D})$		**Goodness measure** $\min_\theta goodness^2(\mathcal{D})$		**Goodness measure** $\max_\theta goodness^3(\mathcal{D})$	
$\mathcal{D}$	δ_{opt}	$\frac{\lvert\{\delta_1,\cdots,\delta_{stop}\}\rvert}{\lvert\{\delta_1,\cdots,\delta_m\}\rvert}$	δ_{opt}	$\frac{\lvert\{\delta_1,\cdots,\delta_{stop}\}\rvert}{\lvert\{\delta_1,\cdots,\delta_m\}\rvert}$	δ_{opt}	$\frac{\lvert\{\delta_1,\cdots,\delta_{stop}\}\rvert}{\lvert\{\delta_1,\cdots,\delta_m\}\rvert}$	δ_{opt}	$\frac{\lvert\{\delta_1,\cdots,\delta_{stop}\}\rvert}{\lvert\{\delta_1,\cdots,\delta_m\}\rvert}$	δ_{opt}	$\frac{\lvert\{\delta_1,\cdots,\delta_{stop}\}\rvert}{\lvert\{\delta_1,\cdots,\delta_m\}\rvert}$
Figure 9 7×7 [20]	0.444	$\frac{10}{10}$	0.444	$\frac{5}{10}$	0.444	$\frac{10}{10}$	0.065	$\frac{10}{10}$	0.086	$\frac{10}{10}$
Figure 10 12×12 *Zoo* [46]	0.559	$\frac{48}{48}$	0.559	$\frac{5}{48}$	0.559	$\frac{48}{48}$	0.348	$\frac{48}{48}$	0.348	$\frac{48}{48}$
Figure 11 4×4 *Mushroom*[46]	0.41	$\frac{4}{4}$	0.41	$\frac{2}{4}$	0.365	$\frac{4}{4}$	0.365	$\frac{4}{4}$	0.41	$\frac{4}{4}$
Figure 12 6×6 *Iris*[46]	0.3	$\frac{6}{6}$	0.3	$\frac{3}{6}$	0.3	$\frac{6}{6}$	0.3	$\frac{6}{6}$	0.3	$\frac{6}{6}$
Figure 13 6×6 *Zoo* & *Mushroom*[46]	0.384	$\frac{8}{8}$	0.384	$\frac{7}{8}$	0.384	$\frac{8}{8}$	0.384	$\frac{8}{8}$	0.384	$\frac{8}{8}$
Figure 14 4×4 [39]	0.429	$\frac{6}{6}$	0.429	$\frac{3}{6}$	0.25	$\frac{6}{6}$	0.25	$\frac{6}{6}$	0.429	$\frac{6}{6}$
Figure 15 10×10 T10I4D100K [49]	0.846	$\frac{31}{31}$	0.846	$\frac{4}{31}$	0.737	$\frac{31}{31}$	0.737	$\frac{31}{31}$	0.737	$\frac{31}{31}$

Table 11. F-measure [60,61], precision [60,61] and recall [60,61] reached by the compared clustering algorithms in [22], [23], [21] and our proposed algorithm on the datasets shown in Figures 9-15. Notice that our algorithm gets the best scores for all the datasets.

Dataset	**F-measure[60,61]**				**precision[60,61]**				**recall[60,61]**			
	Proposed Algo	**Algo[22]**	**Algo[23]**	**Algo[21]**	**Proposed Algo**	**Algo[22]**	**Algo[23]**	**Algo[21]**	**Proposed Algo**	**Algo[22]**	**Algo[23]**	**Algo[21]**
Figure 9 7×7 [20]	1	1	0.44	0.8	1	1	0.28	0.66	1	1	1	1
Figure 10 12×12 *Zoo* [46]	1	1	0.14	0.14	1	1	0.075	0.075	1	1	1	1
Figure 11 4×4 *Mushroom*[46]	1	0.66	0.66	1	1	0.5	0.5	1	1	1	1	1
Figure 12 6×6 *Iris*[46]	1	1	1	1	1	1	1	1	1	1	1	1
Figure 13 6×6 *Zoo* & *Mushroom*[46]	1	1	1	1	1	1	1	1	1	1	1	1
Figure 14 4×4 [39]	1	0.66	0.66	1	1	0.5	0.5	1	1	1	1	1
Figure 15 10×10 T10I4D100K [49]	1	0.16	0.16	0.16	1	0.088	0.088	0.088	1	1	1	1

Table 12. Rand index [62] and Jaccard index [63] reached by the clustering algorithms in [22], [23], [21] and our proposed algorithm on the datasets shown in Figures 9-15. Notice that our algorithm gets the best scores for all the datasets.

Dataset	**Rand[62]**				**Jaccard[63]**			
	Proposed Algo	**Algo[22]**	**Algo[23]**	**Algo[21]**	**Proposed Algo**	**Algo[22]**	**Algo[23]**	**Algo[21]**
Figure 9 7×7 [20]	1	1	0.28	0.85	1	1	0.28	0.66
Figure 10 12×12 *Zoo* [46]	1	1	0.075	0.075	1	1	0.075	0.075
Figure 11 4×4 *Mushroom*[46]	1	0.5	0.5	1	1	0.5	0.5	1
Figure 12 6×6 *Iris*[46]	1	1	1	1	1	1	1	1
Figure 13 6×6 *Zoo* & *Mushroom*[46]	1	1	1	1	1	1	1	1
Figure 14 4×4 [39]	1	0.5	0.5	1	1	0.5	0.5	1
Figure 15 10×10 T10I4D100K [49]	1	0.088	0.088	0.088	1	0.088	0.088	0.088

Table 13. Summary of the average running times and the average clustering errors $\overline{\mathcal{E}(\mathcal{P},\mathcal{Q})}$ (21) for the proposed algorithm, BestDatabaseClustering [22] and GDMDBClustering [25] (with three different values for the learning rate η) on the random samples described in Table 7.

Experiment (Figure)	Proposed Algo		BestDatabaseClustering[22]		GDMDBClustering[25]	
	Average Running Time	Average Clustering Error	Average Running Time	Average Clustering Error	Average Running Time	Average Clustering Error
Figure 18 $\eta = 0.0005$	6.367	0.285	47.208	0.936	28.479	0.285
Figure 16 $\eta = 0.001$	6.367	0.285	47.208	0.936	14.825	0.285
Figure 17 $\eta = 0.002$	6.367	0.285	47.208	0.936	7.305	0.290

Table 14. Statistical test results for the measurements obtained by all the compared clustering algorithms in Figure 16.

Algorithm	Measurements									
	Running Time					Clustering Error				
	Average	SD	Var	stat	p-value	Average	SD	Var	stat	p-value
Proposed Algo	6.367	3.018	9.107			0.285	0.080	0.006		
BestDatabaseClustering[22]	47.208	27.537	758.313	135.707	$3.40e-30$	0.936	0.066	0.004	150	$2.67e-33$
GDMDBClustering [25] ($\eta = 0.001$)	14.825	1.743	3.037			0.285	0.080	0.006		

Table 15. Statistical test results for the measurements obtained by all the compared clustering algorithms in Figure 17.

Algorithm	Measurements									
	Running Time					Clustering Error				
	Average	SD	Var	stat	p-value	Average	SD	Var	stat	p-value
Proposed Algo	6.367	3.018	9.107			0.285	0.080	0.006		
BestDatabaseClustering[22]	47.208	27.537	758.313	121.62	$3.88e-27$	0.936	0.066	0.004	131	$3.56e-29$
GDMDBClustering [25] ($\eta = 0.002$)	7.305	1.766	3.118			0.290	0.086	0.007		

Table 16. Statistical test results for the measurements obtained by all the compared clustering algorithms in Figure 18.

Algorithm	Measurements									
	Running Time					Clustering Error				
	Average	SD	Var	stat	p-value	Average	SD	Var	stat	p-value
Proposed Algo	6.367	3.018	9.107			0.285	0.080	0.006		
BestDatabaseClustering[22]	47.208	27.537	758.313	118.90	$1.51e-26$	0.936	0.066	0.004	150	$2.67e-33$
GDMDBClustering [25] ($\eta = 0.0005$)	28.479	4.655	21.669			0.285	0.080	0.006		

Table 17. Contingency matrix showing the categories in pairing clustered databases.

Clustering	**Predicted clusters**	
	Pairs *in* $\mathcal{P}$	**Pairs *not in* $\mathcal{P}$**
Actual clusters		
Pairs *in* $\mathcal{Q}$	$a := \lvert Pairs_{\mathcal{Q}} \cap Pairs_{\mathcal{P}} \rvert$ (True Positive)	$b := \lvert Pairs_{\mathcal{Q}} \setminus Pairs_{\mathcal{P}} \rvert$ (False Negative)
Pairs *not in* $\mathcal{Q}$	$c := \lvert Pairs_{\mathcal{P}} \setminus Pairs_{\mathcal{Q}} \rvert$ (False Positive)	$d :=$ Pairs *in none* (True Negative)

Table 18. Pair counting measures used for clustering assessment and comparison.

precision [60,61]	**recall** [60,61]	**F-measure** [60,61]	**Rand** [62]	**Jaccard** [63]
$\frac{(a)}{(a)+(c)}$	$\frac{(a)}{(a)+(b)}$	$\frac{2(a)}{2(a)+(b)+(c)}$	$\frac{(a)+(d)}{(a)+(b)+(c)+(d)}$	$\frac{(a)}{(a)+(b)+(c)}$

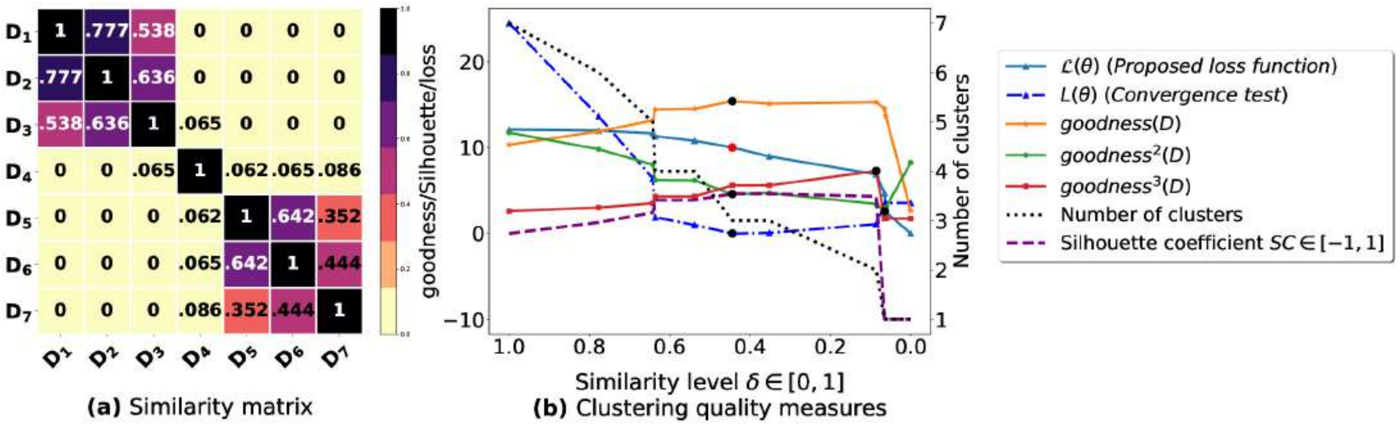


Figure 9. (a): represents a similarity matrix between 7 databases from [25]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p ($p = 1 \cdots 7$) under a threshold $\alpha = 0.42$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)}$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 5$ and the number of generated clusters.

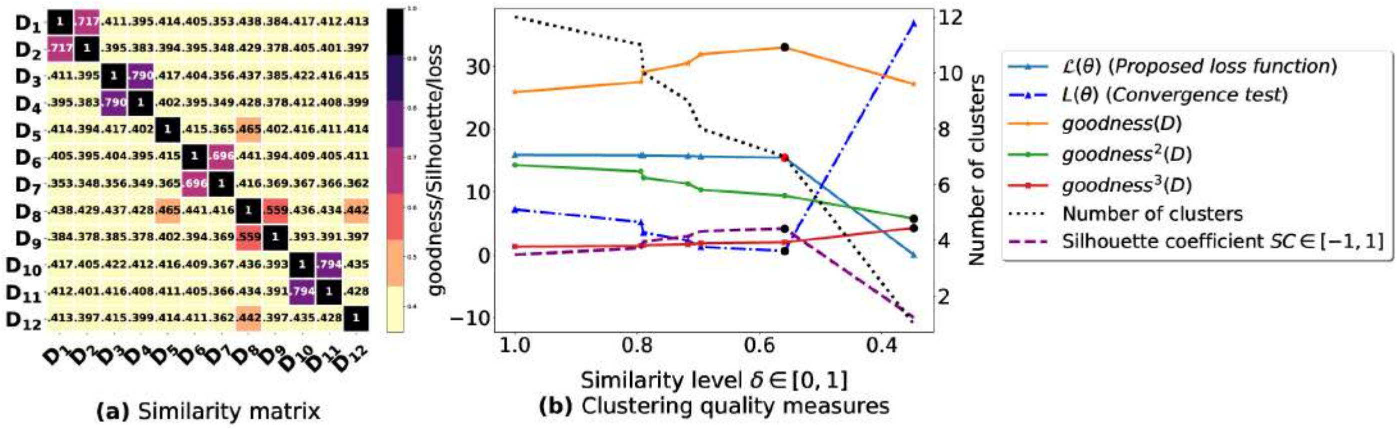


Figure 10. (a): represents a similarity matrix between 12 databases partitioned from Zoo dataset [46]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p ($p = 1 \cdots 12$) under a threshold $\alpha = 0.5$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)} \div 10$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 2$ and the number of generated clusters.

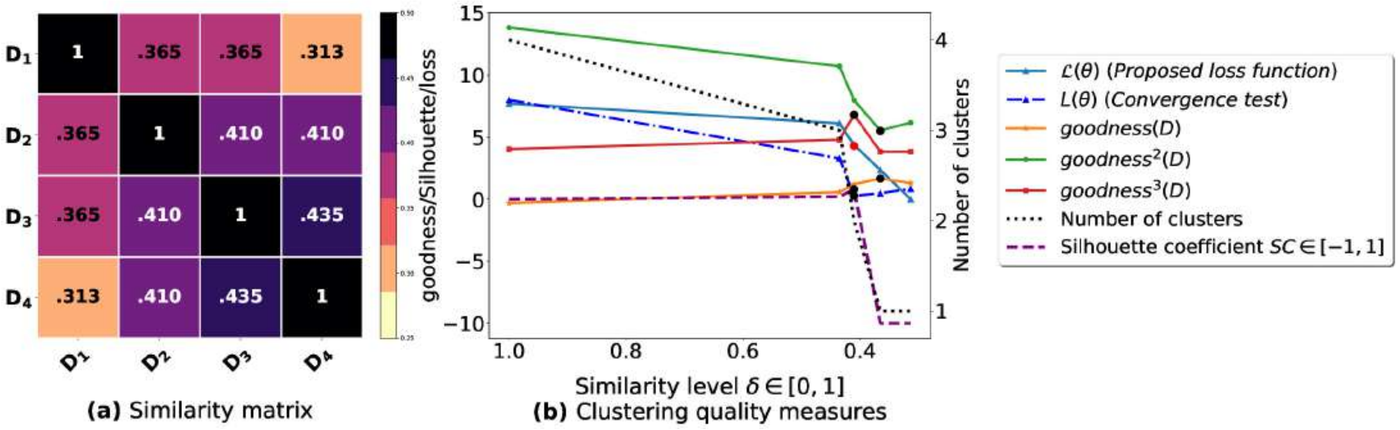


Figure 11. **(a)**: represents a similarity matrix between 4 databases partitioned from Mushroom dataset [46] without applying the fuzziness reduction model in Figure 2. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p ($p = 1 \cdots 4$) under a threshold $\alpha = 0.5$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)}$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 10$ and the number of generated clusters.

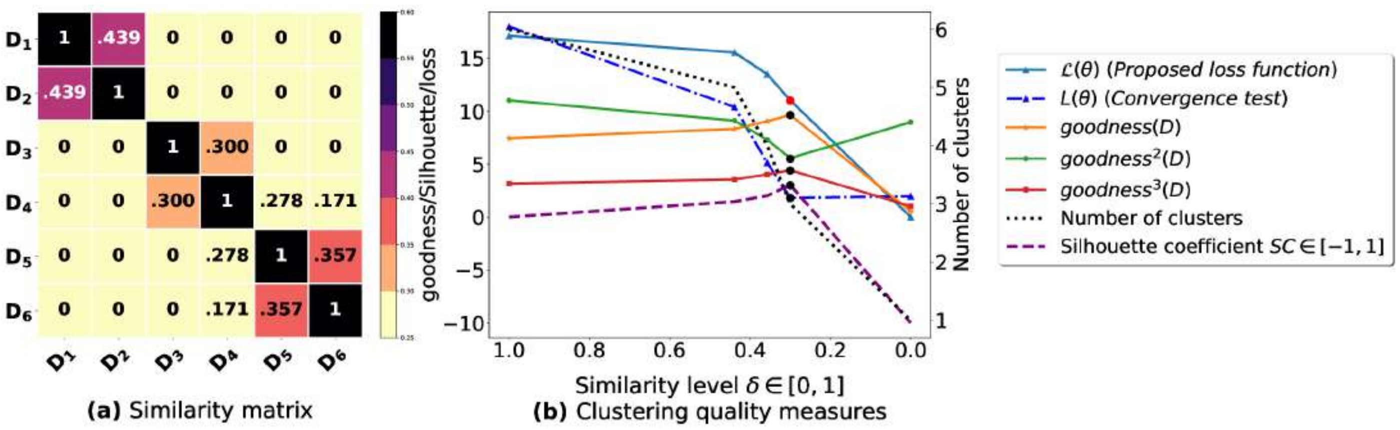


Figure 12. **(a)**: represents a similarity matrix between 6 databases partitioned from Iris dataset [46]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p ($p = 1 \cdots 6$) under a threshold $\alpha = 0.2$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)}$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 10$ and the number of generated clusters.

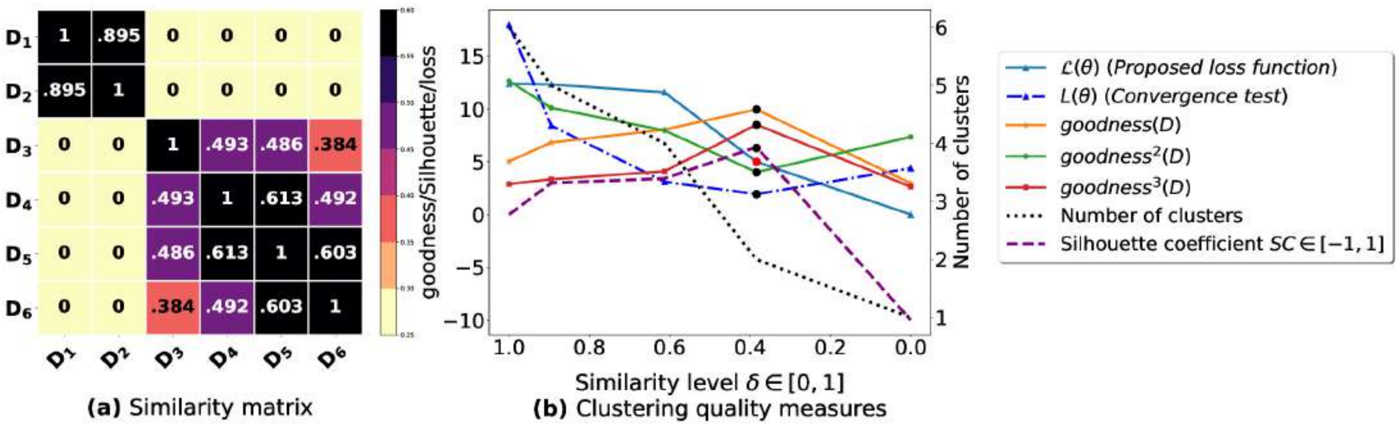


Figure 13. **(a)**: represents a similarity matrix between 6 databases including 4 databases partitioned from the real dataset Mushroom [46] and 2 databases partitioned from the real dataset Zoo [46]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p $(p = 1 \cdots 6)$ under a threshold $\alpha = 0.5$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)}$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 10$ and the number of generated clusters.

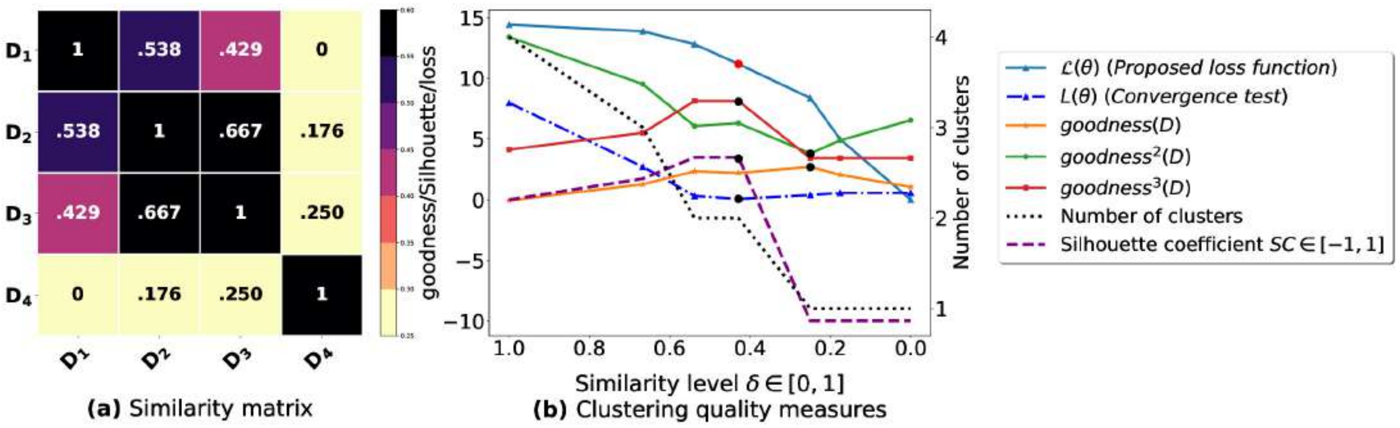


Figure 14. **(a)**: represents a similarity matrix between 4 databases from [39]. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)}$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 10$ and the number of generated clusters.

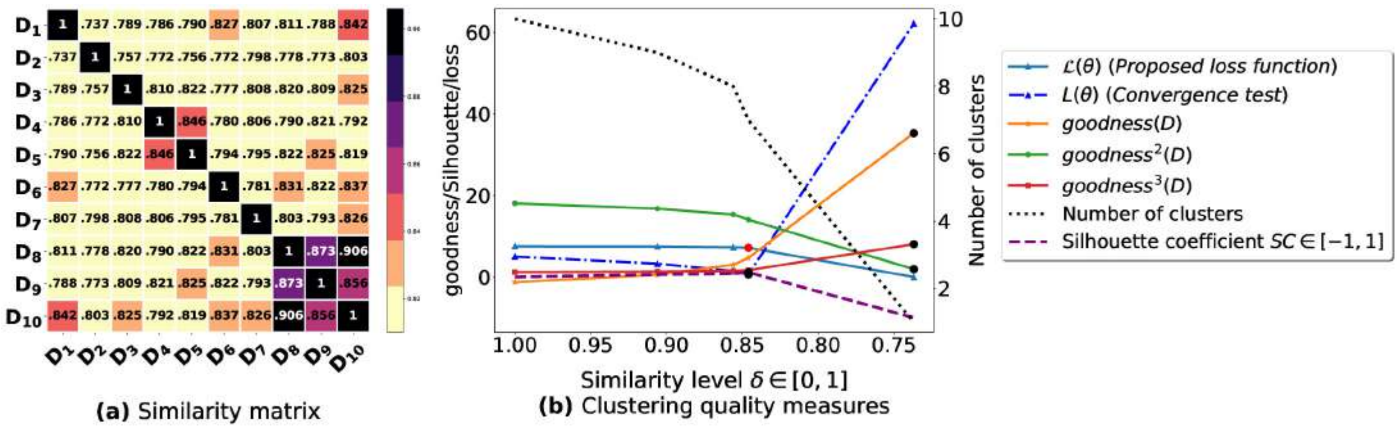


Figure 15. **(a)**: represents a similarity matrix between 10 databases partitioned from T10I4D100K [49]. We note that (a) is built by calling *sim* (2) on the frequent itemsets (FIs) mined from D_p $(p = 1 \cdots 10)$ under a threshold $\alpha = 0.03$. **(b):** represents the graph plots corresponding to $\boldsymbol{goodness^3(\mathcal{D})} \times 10$ [21], $\boldsymbol{goodness^2(\mathcal{D})} \times 10$ [23], $\boldsymbol{goodness(\mathcal{D})}$ [20], the Silhouette Coefficient $\boldsymbol{SC} \times 10$ [43], our convergence test function $\boldsymbol{L(\theta)} \div 10$, our proposed loss function $\boldsymbol{\mathcal{L}(\theta)} \times 10$ and the number of generated clusters.

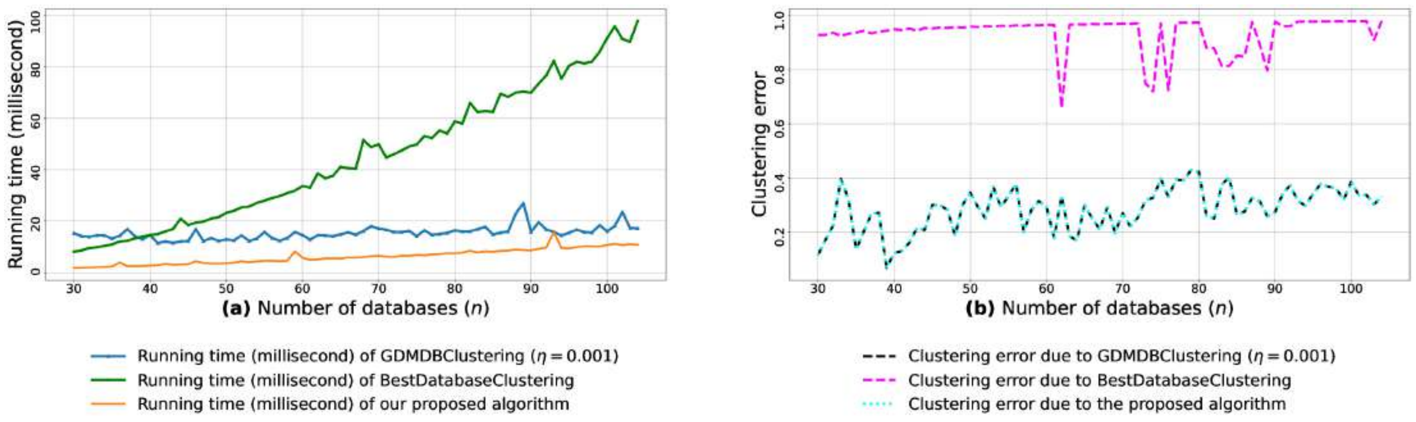


Figure 16. (a): represents the running times corresponding to GDMDBClustering [25] (with a learning rate $\eta = 0.001$), BestDatabaseClustering [22] and the proposed algorithm obtained when executed on $n = 30 \cdots 120$ isotropic Gaussian blobs generated using scikit-learn generator [50]. **(b)**: represents the clustering error graphs (20) due to GDMDBClustering [25], BestDatabaseClustering [22] and the proposed algorithm.

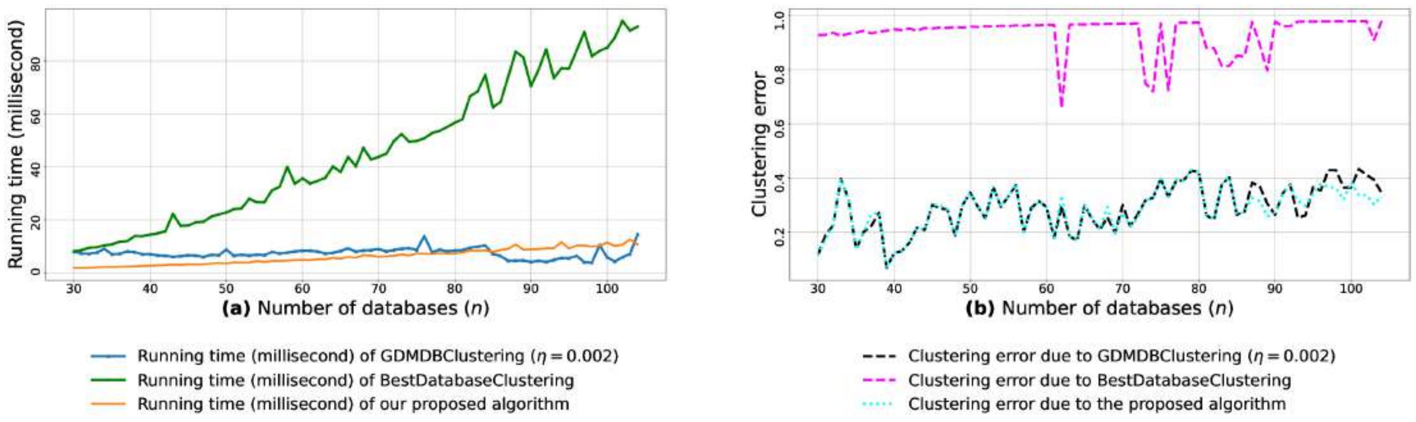


Figure 17. (a): represents the running times corresponding to GDMDBClustering [25] (with a learning rate $\eta = 0.002$), BestDatabaseClustering [22] and the proposed algorithm obtained when executed on $n = 30 \cdots 120$ isotropic Gaussian blobs generated using scikit-learn generator [50]. **(b)**: represents the clustering error graphs (20) due to GDMDBClustering [25], BestDatabaseClustering [22] and the proposed algorithm.

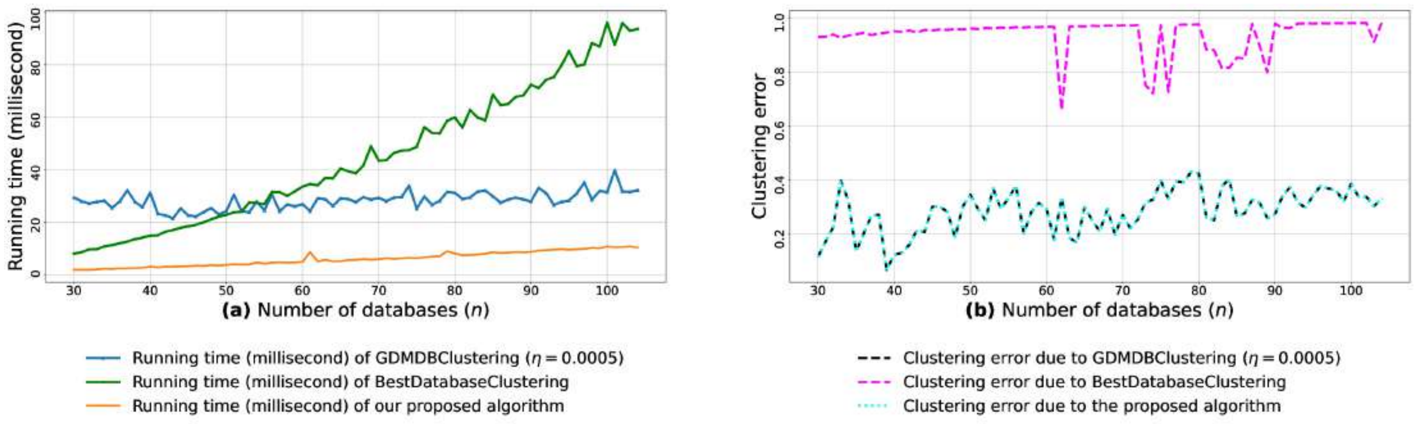


Figure 18. (a): represents the running times corresponding to GDMDBClustering [25] (with a learning rate $\eta = 0.0005$), BestDatabaseClustering [22] and the proposed algorithm obtained when executed on $n = 30 \cdots 120$ isotropic Gaussian blobs generated using scikit-learn generator [50]. **(b)**: represents the clustering error graphs (20) due to GDMDBClustering [25], BestDatabaseClustering [22] and the proposed algorithm.

5. Conclusions

An improved similarity-based clustering algorithm for multi-database mining was proposed in this paper. Unlike the previous works, our algorithm requires fewer upper-bounded iterations to minimize a convex clustering quality measure. In addition, we have proposed a preprocessing layer prior to clustering where the pairwise similarities between multiple databases are first adjusted to reduce their fuzziness. This will help the clustering process to be more precise and less confused in discriminating between the different database clusters. To assess the performance of our algorithm, we have conducted several experiments on real and synthetic datasets. Compared with the existing clustering algorithms for multi-database mining, our algorithm achieved the best performance in terms of accuracy and running time. In this paper, we have used the frequent itemsets mined from each transaction database as feature sets to compute the pairwise similarities between the multiple databases. However, when the sizes of these input vectors become large, building the similarity matrix will increase the CPU overhead drastically. Moreover, the existence of some noisy frequent itemsets (FIs) may largely influence how databases are clustered together. In future work, we will investigate the impact of compressing the size of the FIs into a latent variable represented in a lower dimensional space with discriminative features. Practically, reconstituting the input vectors from the embedding space using deep auto-encoders and non-linear dimensionality reduction techniques, such as T-SNE (t-Distributed Stochastic Neighbor Embedding) and UMAP (Uniform Manifold Approximation and Projection), will force the removal of the noisy features present in the input data while keeping only the meaningful discriminative ones. Consequently, this may help improve the accuracy and running time of the clustering algorithm. Last but not least, in order to design a parallel version of the proposed algorithm, we will study and explore some high-performance computing tools, such as MapReduce and Spark, as an attempt to improve the clustering performance for multi-database mining.

Author Contributions: Conceptualization, S.M.; methodology, S.M.; software, S.M.; validation, S.M.; formal analysis, S.M.; investigation, S.M.; resources, Y.W. and D.W.; data curation, S.M.; writing—original draft preparation, S.M.; writing—review and editing, S.M., Y.W. and D.W.; visualization, S.M.; supervision, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All the datasets are available at http://fimi.ua.ac.be/data/ and https://archive.ics.uci.edu/ml/datasets.

Acknowledgments: We would like to thank the anonymous reviewers for their time and their valuable comments.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

FIs	Frequent Itemsets
FIM	Frequent Itemset Mining
MDB	Multiple Databases
MDM	Multi-database Mining
CD	Coordinate Descent
CL	Competitive Learning
BMU	Best Matching Unit
T-SNE	t-Distributed Stochastic Neighbor Embedding
UMAP	Uniform Manifold Approximation and Projection

References

1. Han, J.; Pei, J.; Yin, Y.; Mao, R. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data mining and knowledge discovery* **2004**, *8*, 53–87.
2. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On spectral clustering: Analysis and an algorithm. Advances in neural information processing systems, 2002, pp. 849–856.
3. Johnson, S.C. Hierarchical clustering schemes. *Psychometrika* **1967**, *32*, 241–254.
4. MacQueen, J.; others. Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. Oakland, CA, USA, 1967, Vol. 1, pp. 281–297.
5. Zhang, Y.J.; Liu, Z.Q. Self-splitting competitive learning: a new on-line clustering paradigm. *IEEE Transactions on Neural Networks* **2002**, *13*, 369–380.
6. Yair, E.; Zeger, K.; Gersho, A.; others. Competitive learning and soft competition for vector quantizer design. *IEEE transactions on Signal Processing* **1992**, *40*, 294–309.
7. Hofmann, T.; Buhmann, J.M. Competitive learning algorithms for robust vector quantization. *IEEE Transactions on Signal Processing* **1998**, *46*, 1665–1675.
8. Kohonen, T. *Self-organizing maps*; Vol. 30, Springer Science & Business Media: Berlin Heidelberg New York, 2012.
9. Pal, N.R.; Bezdek, J.C.; Tsao, E.K. Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE transactions on Neural Networks* **1993**, *4*, 549–557.
10. Mao, J.; Jain, A.K. A self-organizing network for hyperellipsoidal clustering (HEC). *Ieee transactions on neural networks* **1996**, *7*, 16–29.
11. Anderberg, M.R. *Cluster analysis for applications: probability and mathematical statistics: a series of monographs and textbooks*; Vol. 19, Academic press, 2014.
12. Aggarwal, C.C.; Reddy, C.K. Data clustering. *Algorithms and Application, Boca Raton: CRC Press* **2014**.
13. Wang, C.D.; Lai, J.H.; Philip, S.Y. NEIWalk: Community discovery in dynamic content-based networks. *IEEE transactions on knowledge and data engineering* **2013**, *26*, 1734–1748.
14. Wang, Z.; Zhang, D.; Zhou, X.; Yang, D.; Yu, Z.; Yu, Z. Discovering and profiling overlapping communities in location-based social networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2013**, *44*, 499–509.
15. Huang, D.; Lai, J.H.; Wang, C.D.; Yuen, P.C. Ensembling over-segmentations: From weak evidence to strong segmentation. *Neurocomputing* **2016**, *207*, 416–427.
16. Shi, J.; Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **2000**, *22*, 888–905.
17. Zhao, Q.; Wang, C.; Wang, P.; Zhou, M.; Jiang, C. A novel method on information recommendation via hybrid similarity. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2016**, *48*, 448–459.
18. Symeonidis, P. ClustHOSVD: Item recommendation by combining semantically enhanced tag clustering with tensor HOSVD. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2015**, *46*, 1240–1251.
19. Rafailidis, D.; Daras, P. The TFC model: Tensor factorization and tag clustering for item recommendation in social tagging systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2012**, *43*, 673–688.
20. Adhikari, A.; Adhikari, J. Clustering Multiple Databases Induced by Local Patterns. In *Advances in knowledge discovery in databases*; Springer: Switzerland, 2015; pp. 305–332.
21. Liu, Y.; Yuan, D.; Cuan, Y. Completely Clustering for Multi-databases Mining. *Journal of Computational Information Systems* **2013**.
22. Miloudi, S.; Hebri, S.A.R.; Khiat, S. Contribution to Improve Database Classification Algorithms for Multi-Database Mining. *Journal of Information Processing Systems* **2018**, *14*, 709–726.
23. Tang, H.; Mei, Z. A Simple Methodology for Database Clustering. The 5th International Conference on Computer Engineering and Networks. SISSA Medialab, 2015, Vol. 259, p. 019.
24. Wang, R.; Ji, W.; Liu, M.; Wang, X.; Weng, J.; Deng, S.; Gao, S.; Yuan, C.a. Review on mining data from multiple data sources. *Pattern Recognition Letters* **2018**, *109*, 120–128.
25. Miloudi, S.; Wang, Y.; Ding, W. A Gradient-Based Clustering for Multi-Database Mining. *IEEE Access* **2021**, *9*, 11144–11172. doi:10.1109/ACCESS.2021.3050404.
26. Miloudi, S.; Wang, Y.; Ding, W. An Optimized Graph-based Clustering for Multi-database Mining. 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), 2020, pp. 807–812. doi:10.1109/ICTAI50040.2020.00128.
27. Zhang, S.; Zaki, M.J. Mining Multiple Data Sources: Local Pattern Analysis. *Data Mining and Knowledge Discovery* **2006**, *12*, 121–125.
28. Adhikari, A.; Rao, P.R. Synthesizing heavy association rules from different real data sources. *Pattern Recognition Letters* **2008**, *29*, 59–71.
29. Adhikari, A.; Adhikari, J. *Advances in knowledge discovery in databases*; Springer: Switzerland, 2015.
30. Adhikari, A.; Jain, L.C.; Prasad, B. A State-of-the-Art Review of Knowledge Discovery in Multiple Databases. *Journal of Intelligent Systems* **2017**, *26*, 23–34.
31. Zhang, S.; Zhang, C.; Wu, X. Identifying Exceptional Patterns. *Knowledge discovery in multiple databases* **2004**, pp. 185–195.
32. Zhang, S.; Zhang, C.; Wu, X. Identifying High-vote Patterns. *Knowledge discovery in multiple databases* **2004**, pp. 157–183.

33. Ramkumar, T.; Srinivasan, R. Modified algorithms for synthesizing high-frequency rules from different data sources. *Knowledge and information systems* **2008**, *17*, 313–334.
34. Djenouri, Y.; Lin, J.C.W.; Nørvåg, K.; Ramampiaro, H. Highly efficient pattern mining based on transaction decomposition. 2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 2019, pp. 1646–1649.
35. Savasere, A.; Omiecinski, E.R.; Navathe, S.B. An efficient algorithm for mining association rules in large databases. Technical Report GIT-CC-95-04, Georgia Institute of Technology, 1995.
36. Zhang, S.; Wu, X. Large scale data mining based on data partitioning. *Applied Artificial Intelligence* **2001**, *15*, 129–139.
37. Zhang, C.; Liu, M.; Nie, W.; Zhang, S. Identifying Global Exceptional Patterns in Multi-database Mining. *IEEE Intell. Informatics Bull.* **2004**, *3*, 19–24.
38. Zhang, S.; Zhang, C.; Yu, J.X. An efficient strategy for mining exceptions in multi-databases. *Information Sciences* **2004**, *165*, 1–20.
39. Wu, X.; Zhang, C.; Zhang, S. Database classification for multi-database mining. *Information Systems* **2005**, *30*, 71–88.
40. Li, H.; Hu, X.; Zhang, Y. An Improved Database Classification Algorithm for Multi-database Mining. Frontiers in Algorithmics. Springer, Berlin, Heidelberg, 2009, Lecture Notes in Computer Science, pp. 346–357.
41. Na, S.; Xumin, L.; Yong, G. Research on k-means clustering algorithm: An improved k-means clustering algorithm. 2010 Third International Symposium on intelligent information technology and security informatics. IEEE, 2010, pp. 63–67.
42. Selim, S.Z.; Ismail, M.A. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on pattern analysis and machine intelligence* **1984**, pp. 81–87.
43. Rousseeuw, P.J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* **1987**, *20*, 53–65.
44. Kaufman, L.; Rousseeuw, P.J. *Finding groups in data: an introduction to cluster analysis*; Vol. 344, John Wiley & Sons, 2009.
45. De Luca, A.; Termini, S. A Definition of a Nonprobabilistic Entropy in the Setting of Fuzzy Sets Theory. In *Readings in Fuzzy Sets for Intelligent Systems*; Dubois, D.; Prade, H.; Yager, R.R., Eds.; Morgan Kaufmann, 1993; pp. 197–202. doi:https://doi.org/10.1016/B978-1-4832-1450-4.50020-1.
46. Center for Machine Learning and Intelligent Systems. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/. Accessed 10 October 2020.
47. Boyd, S.; Vandenberghe, L. *Convex optimization*; Cambridge university press, 2004.
48. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. Data structures for disjoint sets. *Introduction to Algorithms* **2009**, pp. 498–524.
49. IBM Almaden Quest research group. Frequent Itemset Mining Dataset Repository. http://fimi.ua.ac.be/data/. Accessed 10 October 2020.
50. Thirion, G. Varoquaux, A. Gramfort, V. Michel, O. Grisel, G. Louppe, J. Nothman. Scikit-learn: sklearn.datasets.makeblobs. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html. Accessed 10 October 2020.
51. Alexandre Gramfort, Mathieu Blondel , Olivier Grisel, Andreas Mueller, Eric Martin, Giorgio Patrini and Eric Chang. Scikit-learn: sklearn.preprocessing.MinMaxScaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Accessed 10 October 2020.
52. Friedman, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics* **1940**, *11*, 86–92.
53. Meilă, M. Comparing clusterings: an axiomatic view. Proceedings of the 22nd international conference on Machine learning, 2005, pp. 577–584.
54. Vinh, N.X.; Epps, J.; Bailey, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research* **2010**, *11*, 2837–2854.
55. Günnemann, S.; Färber, I.; Müller, E.; Assent, I.; Seidl, T. External evaluation measures for subspace clustering. Proceedings of the 20th ACM international conference on Information and knowledge management, 2011, pp. 1363–1372.
56. Banerjee, A.; Krumpelman, C.; Ghosh, J.; Basu, S.; Mooney, R.J. Model-based overlapping clustering. Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005, pp. 532–537.
57. Pfitzner, D.; Leibbrandt, R.; Powers, D. Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems* **2009**, *19*, 361–394.
58. Achtert, E.; Goldhofer, S.; Kriegel, H.P.; Schubert, E.; Zimek, A. Evaluation of clusterings–metrics and visual support. 2012 IEEE 28th International Conference on Data Engineering. IEEE, 2012, pp. 1285–1288.
59. Shafiei, M.; Milios, E. Model-based overlapping co-clustering. Proceeding of SIAM Conference on Data Mining, 2006.
60. Chinchor, N. MUC-4 evaluation metrics in Proc. of the Fourth Message Understanding Conference 22–29, 1992.
61. Mei, Q.; Radev, D. Information retrieval. In *The Oxford Handbook of Computational Linguistics 2nd edition*; 1979.
62. Rand, W.M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association* **1971**, *66*, 846–850.
63. Jaccard, P. Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines. *Bull Soc Vaudoise Sci Nat* **1901**, *37*, 241–272.