Design analysis and implementation of efficient parameter free algorithm for high quality homogeneous clusters in data mining applications

Prasad S.Halgaonkar[†], Vijay M.Wadhai[††] and A.D.Potgantwar[†††]

[†] M.Tech-II (Comp Science and Engg), WCE, Sangli INDIA
[††]Prof. & Dean of Research, MITSOT, MAE, Pune INDIA
[†††]Faculty of Computer Engineering, SITRC Nashik INDIA

Abstract
A new algorithm for clustering high-dimensional categorical data is proposed and implemented by us. Our algorithm is parameter-free, fully-automatic and is based on a two-phase iterative procedure. In the first phase, cluster assignments are given, and a new cluster is added to the partition by identifying and splitting a low-quality cluster. Second phase attempts to optimize clusters. This algorithm is parametric to cluster quality in terms of homogeneity. We show how a suitable notion of cluster homogeneity can be defined in the context of high-dimensional categorical data, from which an effective instance of the proposed clustering scheme immediately follows. Our experiments carried out on real data shows that the devised algorithm achieves optimal results in terms of compactness and separation.

Index Terms
Clustering, high-dimensional categorical data, information search and retrieval.

I. INTRODUCTION

Clustering is an unsupervised classification technique. A set of unlabeled objects are grouped into meaningful clusters [1] [2], such that the groups formed are homogeneous and neatly separated. Challenges for clustering categorical data are: 1) Lack of ordering of the domains of the individual attributes. 2) Scalability to high dimensional data in terms of effectiveness and efficiency. High-dimensional categorical data such as market-basket has records containing large number of attributes. 3) Dependency on parameters. Setting of many input parameters is required for many of the clustering techniques which lead to many critical aspects.

Parameters are useful in many ways. Parameters support requirements such as efficiency, scalability, and flexibility. For proper tuning of parameters a lot of effort is required. As number of parameters increases, the problem of parameter tuning also increases. Algorithm should have as less parameters as possible. If the algorithm is automatic it helps to find accurate clusters. An automatic approach technique searches huge amounts of high-dimensional data such that it is effective and rapid which is not possible for human expert. A parameter free approach is based on decision tree learning, which is implemented by top-down divide-and-conquer strategies. The above mentioned problems have been tackled separately, and specific approaches are proposed in the literature, which does not fit the whole framework. The main objective of this paper is to face the three issues in a unified framework. We look forward to an algorithmic technique that is capable of automatically detecting the underlying interesting structure (when available) on high-dimensional categorical data.

We present Efficient Parameter Free (EPF), a new approach to clustering high-dimensional categorical data that scales to processing large volumes of such data in terms of both effectiveness and efficiency. Given an initial data set, it searches for a partition, which improves the overall purity. The algorithm is not dependent on any data-specific parameter (such as the number of clusters or occurrence thresholds for frequent attribute values). It is intentionally left parametric to the notion of purity, which allows for adopting the quality criterion that best meets the goal of clustering. Section 2 reviews some of the related work carried out on transactional data, high dimensional data and high dimensional categorical data. Section 3 provides background information on the clustering of high dimensional categorical data (EPF algorithm). Section 4 describes implementation results of EPF algorithm. Section 5 concludes the paper and draws direction to future work.

II. RELATED WORK

In current literature, many approaches are given for clustering categorical data. Most of these techniques suffer from two main limitations, 1) their dependency on a set of parameters whose proper tuning is required and 2) their lack of scalability to high dimensional data. Most of the approaches are unable to deal with the above features and in giving a good strategy for tuning the parameters.

Many distance-based clustering algorithms [2], [3] are proposed for transactional data. But traditional clustering techniques have the curse of dimensionality and the

Manuscript received February 5, 2010
Manuscript revised February 20, 2010

sparseness issue when dealing with very high-dimensional data such as market-basket data or Web sessions. For example, the K-Means algorithm has been adopted by replacing the cluster mean with the more robust notion of cluster medoid [3] (that is, the object within the cluster with the minimal distance from the other points) or the attribute mode [4]. However, the proposed extensions are inadequate for large values of m: Gozzi et al. [5] describe such inadequacies in detail and propose further extensions to the K-Means scheme, which fit transactional data. Unfortunately, this approach reveals to be parameter laden. When the number of dimensions is high, distance-based algorithms do not perform well. Indeed, several irrelevant attributes might distort the dissimilarity between tuples. Although standard dimension reduction techniques [6] can be used for detecting the relevant dimensions, these can be different for different clusters, thus invalidating such a preprocessing task. Several clustering techniques have been proposed, which identify clusters in subspaces of maximum dimensionality (see [7] for a survey). Though most of these approaches were defined for numerical data, some recent works [8], [9] also consider subspace clustering for categorical data.

A different point of view about (dis)similarity is provided by the ROCK algorithm [27]. The core of the approach is an agglomerative hierarchical clustering procedure based on the concepts of neighbors and links. For a given tuple x, a tuple y is a neighbor of x if the Jaccard similarity $J(x, y)$ between them exceeds a prespecified threshold Θ. The algorithm starts by assigning each tuple to a singleton cluster and merges clusters on the basis of the number of neighbors (links) that they share until the desired number of clusters is reached. ROCK is robust to high-dimensional data. However, the dependency of the algorithm to the parameter Θ makes proper tuning difficult.

Categorical data clusters are considered as dense regions within the data set. The density is related to the frequency of particular groups of attribute values. The higher the frequency of such groups the stronger the clustering. Preprocessing the data set is carried by extracting relevant features (frequent patterns) and discovering clusters on the basis of these features. There are several approaches accounting for frequencies. As an example, Yang et al. [10] propose an approach based on histograms: The goodness of a cluster is higher if the average frequency of an item is high, as compared to the number of items appearing within a transaction. The algorithm is particularly suitable for large high-dimensional databases, but it is sensitive to a user defined parameter (the repulsion factor), which weights the importance of the compactness/sparseness of a cluster. Other approaches [11], [12], [13], [9] extend the computation of frequencies to frequent patterns in the underlying data set. In particular, in [11], [12], each transaction is seen as a relation over some sets of items, and a hyper-graph model is used for representing these relations. Hyper-graph partitioning algorithms can hence be used for obtaining item/transaction clusters.

The CLICKS algorithm proposed in [9] encodes a data set into a weighted graph structure $G(N, E)$, where the individual attribute values correspond to weighted vertices in N, and two nodes are connected by an edge if there is a tuple where the corresponding attribute values co-occur. The algorithm starts from the observation that clusters correspond to dense (that is, with frequency higher than a user-specified threshold) maximal k-partite cliques and proceeds by enumerating all maximal k-partite cliques and checking their frequency. A crucial step is the computation of strongly connected components, that is, pairs of attribute values whose co-occurrence is above the specified threshold. For large values of m (or, more generally, when the number of dimensions or the cardinality of each dimension is high), this is an expensive task, which invalidates the efficiency of the approaches. In addition, technique depends upon a set of parameters, whose tuning can be problematic in practical cases.

Categorical clustering can be tackled by using information-theoretic principles and the notion of entropy to measure closeness between objects. The basic intuition is that groups of similar objects have lower entropy than those of dissimilar ones. The COOLCAT algorithm [14] proposes a scheme where data objects are processed incrementally, and a suitable cluster is chosen for each tuple such that at each step, the entropy of the resulting clustering is minimized. The scaLable InforMation BOttleneck (LIMBO) algorithm [15] also exploits a notion of entropy to catch the similarity between objects and defines a clustering procedure that minimizes the information loss. The algorithm builds a Distributional Cluster Features (DCF) tree to summarize the data in k clusters, where each node contains statistics on a subset of tuples. Then, given a set of k clusters and their corresponding DCFs, a scan over the data set is performed to assign each tuple to the cluster exhibiting the closest DCF. The generation of the DCF tree is parametric to a user-defined branching factor and an upper bound on the distance between a leaf and a tuple.

Li and Ma [16] propose an iterative procedure that is aimed at finding the optimal data partition that minimizes an entropy-based criterion. Initially, all tuples reside within a single cluster. Then, a Monte Carlo process is exploited to randomly pick a tuple and assign it to another cluster as a trial step aimed at decreasing the entropy criterion. Updates are retained whenever entropy diminishes. The overall process is iterated until there are no more changes in cluster assignments. Interestingly, the entropy-based criterion proposed here can be derived in the formal framework of probabilistic clustering models. Indeed, appropriate probabilistic models, namely, multinomial [17] and multivariate Bernoulli [18], have

been proposed and shown to be effective. The classical Expectation-Maximization framework [19], equipped with any of these models, reveals to be particularly suitable for dealing with transactional data [20], [21], being scalable both in n and in m. The correct estimation of an appropriate number of mixtures, as well as a proper initialization of all the model parameters, is problematic here.

The problem of estimating the proper number of clusters in the data has been widely studied in the literature. Many existing methods focus on the computation of costly statistics based on the within-cluster dispersion [22] or on cross-validation procedures for selecting the best model [23], [24]. The latter requires an extra computational cost due to a repeated estimation and evaluation of a predefined number of models. More efficient schemes have been devised in [25], [26]. Starting from an initial partition containing a single cluster, the approaches iteratively apply the K-Means algorithm (with k = 2) to each cluster so far discovered. The decision on whether to switch the original cluster with the newly generated sub-clusters is based on a quality criterion, for example, the Bayesian Information Criterion [25], which mediates between the likelihood of the data and the model complexity, or the improvement in the rate of distortion (the variance in the data) of the sub-clusters with respect to the original cluster [26]. The exploitation of the K-Means scheme makes the algorithm specific to low-dimensional numerical data, and proper tuning to high-dimensional categorical data is problematic.

Automatic approaches that adopt the top-down induction of decision trees are proposed in [28], [29], [30]. The approaches differ in the quality criterion adopted, for example reduction in entropy [28], [29] or distance among the prototypes of the resulting clusters [29]. All of these approaches have some of the drawbacks. The scalability on high-dimensional data is poor. Some of the literature that focused on high dimensional categorical data is available in [31], [32].

III. The EPF Algorithm

The key idea of Efficient Parameter Free (EPF) algorithm is to develop a clustering procedure, which has the general sketch of a top-down decision tree learning algorithm. First, start from an initial partition which contains single cluster (the whole data set) and then continuously try to split a cluster within the partition into two sub-clusters. If the sub-clusters have a higher homogeneity in the partition than the original cluster, the original is removed. The sub-clusters obtained by splitting are added to the partition. Split the clusters on the basis of their homogeneity. A function *Quality(C)* measures the degree of homogeneity of a cluster C. Clusters with high intra-homogeneity exhibit high values of Quality.

Let M be set of Boolean attributes such that $M = \{a_1, \ldots, a_m\}$ and a data set $D = \{x_1, x_2, \ldots, x_n\}$ of tuples which is defined on M. $a \in M$ is denoted as an item, and a tuple $x \in D$ as a transaction x. Data sets containing transactions are denoted as transactional data, which is a special case of high-dimensional categorical data. A cluster is a set S which is a subset of D. The size of S is denoted by n_S, and the size of $M_S = \{a | a \in x, x \in S\}$ is denoted by m_S. A partitioning problem is to divide the original collection of data D into a set $P = \{C_1, \ldots, C_k\}$ where each clusters C_j are nonempty. Each cluster contains a group of homogeneous transactions. Clusters where transactions have several items have higher homogeneity than other subsets where transactions have few items. A cluster of transactional data is a set of tuples where few items occur with higher frequency than somewhere else.

Our approach to clustering starts from the analysis of the analogies between a clustering problem and a classification problem. In both cases, a model is evaluated on a given data set, and the evaluation is positive when the application of the model locates fragments of the data exhibiting high homogeneity. A simple rather intuitive and parameter-free approach to classification is based on decision tree learning, which is often implemented through top-down divide and conquers strategies. Here, starting from an initial root node (representing the whole data set), iteratively, each data set within a node is split into two or more subsets, which define new sub-nodes of the original node. The criterion upon which a data set is split (and, consequently, a node is expanded) is based on a quality criterion: choosing the best "discriminating" attribute (that is, the attribute producing partitions with the highest homogeneity) and partitioning the data set on the basis of such attribute. The concept of homogeneity has found several different explanations (for example, in terms of entropy or variance) and, in general, is related to the different frequencies of the possible labels of a target class. The general schema of the EPF algorithm is specified in Fig. 1. The algorithm starts with a partition having a single cluster i.e whole data set (line 1). The central part of the algorithm is the body of the loop between lines 2 and 15. Within the loop, an effort is made to generate a new cluster by 1) choosing a candidate node to split (line 4), 2) splitting the candidate cluster into two sub-clusters (line 5), and (line 3) calculating whether the splitting allows a new partition with better quality than the original partition (lines 6–13). If this is true, the loop can be stopped (line 10), and the partition is updated by replacing the original cluster with the new sub-clusters (line 8). Otherwise, the sub-clusters are discarded, and a new cluster is taken for splitting.

```
GENERATE-CLUSTERS(D)
    Input: A set D ={x1,...,xN} of transactions;
    Output: A partition P = {C1,...,Ck} of clusters;
1.  Let initially P = {D};
2.  repeat
3.      Generate a new cluster C initially empty;
4.      for each cluster Ci ∈ P do
5.          PARTITION-CLUSTERS(Ci,C);
6.          P' ← P U {C};
7.          if Quality(P) < Quality(P') then
8.              P ← P';
9.              STABILIZE-CLUSTERS(P);
10.             break
11.         else
12.             Restore all xj ∈ C into Ci;
13.         end if
14.     end for
15. until no further cluster C can be
    generated
```

Figure 1: Generate Clusters

The generation of a new cluster calls STABILIZE-CLUSTERS in line 9, improves the overall quality by trying relocations among the clusters. Clusters at line 4 are taken in increasing order of quality.

a. *Splitting a Cluster*

A splitting procedure gives a major improvement in the quality of the partition. Choose the attribute that gives the highest improvement in the quality of the partition.

PARTITION-CLUSTER

The PARTITION-CLUSTER algorithm is given in Fig.2. The algorithm continuously evaluates, for each element x $\in C_1 \cup C_2$, to check whether a reassignment increases the homogeneity of the two clusters.

```
PARTITION-CLUSTER(C1,C2)
P1.   repeat
P2.     for all x ∈ C1 U C2 do
P3.         if cluster(x) = C1 then
P4.             Cu ← C1; Cv ← C2;
P5.         else
P6.             Cu ← C2; Cv ← C1;
P7.         end if
P8.         Qi ← Quality(Cu) + Quality(Cv);
P9.         Qs ← Quality(Cu – {x}) + Quality(Cv U {x});
P10.        if Qs > Qi then
P11.            Cu.Remove(x);
P12.            Cv.Insert(x);
P13.        end if
P14.    end for
P15.  until C1 and C2 are stable
```

Figure 2: Partition Cluster

Lines P8 and P9 compute the involvement of *x* to the local quality in two cases: either *x* remains in its original cluster (C_u) or *x* is moved to the other cluster (C_v). If moving *x* gives an improvement in the local quality, then the swapping is done (lines P10–P13). Lines P2–P14 in the algorithm is nested into a main loop: elements are continuously checked for swapping until a convergence is met. The splitting process can be sensitive to the order upon which elements are considered: In the first stage, it could be not convenient to reassign the generic x_i from C_1 to C_2, whereas a convenience in performing the swap can be found after the relocation of some other element x_j. The main loop partly smoothes this effect by repeatedly relocating objects until convergence is met. Better PARTITION-CLUSTER can be made strongly insensitive to the order with which cluster elements are considered. The basic idea is discussed next. The elements that mostly influence the locality effect are either outlier transactions (that is, those containing mainly items, whose frequency within the cluster is rather low) or common transactions (which, dually, contain very frequent items). In the first case, C_2 is unable to attract further transactions, whereas in the second case, C_2 is likely to attract most of the transactions (and, consequently, C_1 will contain outliers).
The key idea is to rank and sort the cluster elements before line P1, which is on the basis of their splitting effectiveness. To this purpose, each transaction x belonging to cluster C can be associated with a weight w(x), which indicates its splitting effectiveness. x is eligible for splitting C if its items allow us to divide C into two homogeneous sub-clusters. In this respect, the Gini index is a natural way to quantify the splitting effectiveness G(a) of the individual attribute value $a \in x$. Precisely, $G(a) = 1 - Pr(a|C)^2 - (1 - Pr(a|C))^2$, where Pr(a|C) denotes the probability of a within C. G(a) is close to its maximum whenever a is present in about half of the transactions of C and reaches its minimum whenever a is unfrequent or common within C. The overall splitting effectiveness of x can be defined by averaging the splitting effectiveness of its constituting items $w(x) = avg_{a \in x}(G(a))$. Once ranked, the elements $x \in C$ can be considered in descending order of their splitting effectiveness at line P2. This guarantees that C_2 is initialized with elements, which do not represent outliers and still are likely to be removed from C_1. This removes the dependency on the initial input order of the data. With decision tree learning, EPF exhibits a preference bias, which is encoded within the notion of homogeneity and can be viewed as the preference for compact clustering trees. Indeed, due to the splitting effectiveness heuristic, homogeneity is enforced by the effects of the Gini index. At each split, this tends to isolate clusters of transactions with mostly frequent attribute values, from which the compactness of the overall clustering tree follows.

```
STABILIZE-CLUSTERS(P)
S1.   repeat
S2.     for all x ∈ D do
S3.       C_pivot ← cluster(x); Q ← Quality(P);
S4.       for all C ∈ P do
S5.         C_pivot.REMOVE(x);
S6.         C.INSERT(x);
S7.         if Quality(P) > Q then
S8.           if C_pivot = Ø then
S9.             P.REMOVE(C_pivot);
S10.          end if
S11.            C_pivot ← C; Q ← Quality(P);
S12.        else
S13.          C_pivot.INSERT(x);
S14.          C.REMOVE(x);
S15.        end if
S16.      end for
S17.    end for
S18.  until P is stable
```

Figure 3: Stabilize Clusters

b. STABILIZE-CLUSTERS

PARTITION-CLUSTER improves the local quality of a cluster. And STABILIZE-CLUSTERS try to increase partition quality. It is carried out by finding the most suitable clusters for each element among the ones which are there in the partition. Fig. 3 shows the pseudo code of the procedure. The central part of the algorithm is a main loop which (lines S2–S17) examines all the available elements. For each element *x*, a pivot cluster is identified, which is the cluster containing *x*. Then, the available clusters are continuously evaluated. The insertion of *x* in the current cluster is done (lines S5–S6), and the updated quality is compared with the original quality. If an improvement is obtained, then the swap is accepted (line S11). The new pivot cluster is the one now containing *x*, and if the removal of *x* makes the old pivot cluster empty, then the old pivot cluster is removed from the partition P. If there is no improvement in quality, *x* is restored into its pivot cluster, and a new cluster is examined. The main loop is iterated until a stability condition for clusters is achieved.

c. *Cluster and Partition Qualities*

AT-DC gives two different quality measures, 1) local homogeneity within a cluster and 2) global homogeneity of the partition. As shown in Fig. 1, it is noticed that partition quality is used for checking whether the insertion of a new cluster is really suitable: it is for maintaining compactness. Cluster quality in procedure PARTITIONCLUSTER is done for good separation..

Cluster quality is known when there is a high degree of intracluster homogeneity and intercluster homogeneity. As given in [35], there is strong relation between intracluster homogeneity and the probability $Pr(a_i|C_k)$ that item a_i appears in a transaction containing in C_k. There is a strong relationship between intercluster separation and Pr(x $\in C_k$, $a_i \in$ x). Cluster homogeneity and separation is computed by relating it to the unity of items within the transactions that it contains. Cluster quality is equal to the combination of the above probability, $\sum_{a \in Mc} \Pr(a|C)\Pr(C|a)\Pr(a)$. The last term is used for weighting the importance of item a in the summation: Essentially, high values from low-frequency items are less relevant than those from high-frequency values. By the Bayes theorem, the above formula is expressed as $\Pr(C) \sum_{a \in Mc} \Pr(a|C)^2$ [33]. Terms $\Pr(a|C)^2$ (relative strength of a within C) and Pr(C) (relative strength of C) work in contraposition. It is easy to compute the gain in strength for each item with respect to the whole data set, that is

$$\textbf{Quality } (C_k) = \Pr(C_k) \sum_{a \in Mck} [\Pr(a|Ck)^2 - \Pr(a|D)^2] \quad \ldots\ldots\ldots (1)$$

Where,

- C_k – cluster
- $Pr(C_k)$ – relative strength of C_k
- *a* ∈ MC_k – *an item*
- $M = \{a_1, \ldots\ldots, a_m\}$ is set of Boolean attributes
- $Pr(a|C_k)$ - relative strength of *a* within C_k
- Pr(a|*D*) - relative strength of *a* within *D*
- $D = \{x_1, \ldots\ldots, x_n\}$ is data set of tuples defined on *M*

$$\textbf{Quality } (C_k) = \frac{n}{N} \sum_{a \in x, x \in C} [\left(\frac{na}{N}\right)^2 - \left(\frac{Na}{N}\right)^2] \quad \ldots\ldots\ldots (2)$$

where na and Na represent the frequencies of a in C and D, respectively. The value of Quality (C_k) is updated as soon as a new transaction is added to C.

IV. RESULTS AND ANALYSIS

Two real-life data sets were evaluated. A description of each data set employed for testing is provided next, together with an evaluation of the EPF performances.

UCI DATASETS [34]

Soybean: The Soybean data set contains 47 records and 35 attributes. Each record contains a class as D1/ D2/ D3

or D4. All 35 attributes are categorical. A detailed result is given in the confusion matrix in table 1. As it is seen, AT-DC found two clusters. It is observed that there is high homogeneity in cluster 1 and 2.

Congressional Votes: The Congressional Votes data set contains a set of US Congressional Voting Records. Each record is one Congressman's votes on 16 issues (for example, education spending, crime, and immigration). All attributes are Boolean ("Yes" or "No" vote), and some contains missing values. A label of "Republican" or "Democrat" is given to each data record. The data set contains 435 records: 168 "Republican" and 267 "Democrat." Table 2 show the results obtained by EPF on the Congressional Votes data set. It is seen that all the three clusters are having high homogeneity.

Table 1: Confusion matrix for soybean

Cluster No.	Classes			
	D1	*D2*	*D3*	*D4*
1	0	0	10	17
2	10	10	0	0

Table 2: Confusion matrix for congressional votes

Cluster No.	No. of Democrats	No. of Republicans
1	143	0
2	0	168
3	124	0

Table 3: Comparison amongst categorical clustering methods (C = Optimal number of clusters)

Dataset	EPF		LIMBO		CLICK		ROCK	
	Timing	C	Timing	C	Timing	C	Timing	C
Soybean	0.07	2	0.62	3	0.94	5	102	2
Cong. Votes	0.18	3	0.81	4	1.06	6	117	3

a. Comparative Analysis

We evaluated EPF versus three main algorithms from the current literature, namely, LIMBO, CLICK and ROCK. CLICK [9] was shown to outperform other approaches adopting a similar hyper-graph partitioning strategy [36], [37] and was chosen for comparison, since this claimed to be capable of dealing with high-dimensional categorical data. ROCK [27] is particularly suitable for market-basket data. LIMBO [15], despite its time complexity, was shown to be quite effective.

Table 3 summarizes the results of the comparison. The results for the LIMBO, ROCK and CLICK algorithms were obtained by performing an accurate tuning of the input parameters: For each data set, different runs were executed for different values of the parameters, and the best results were chosen. The results shown only refer to the run with the best combination of parameters. EPF outperforms LIMBO, ROCK and CLICK in terms of efficiency (i.e. timing). With respect to optimal number of clusters, EPF outperforms LIMBO and CLICK while EPF is comparable to ROCK.

V. CONCLUDING REMARK

The algorithm implemented by us is fully-automatic, parameter-free approach to cluster high-dimensional categorical data. The main advantage of our approach is its capability of avoiding explicit prejudices, expectations, and presumptions on the problem at hand, thus allowing the data itself to speak. This is useful with the problem at hand, where the data is described by several relevant attributes.

A limitation of the our proposed approach is that the underlying notion of cluster quality is not meant for catching conceptual similarities, that is, when distinct values of an attribute are used for denoting the same concept. Probabilities are provided to evaluate cluster homogeneity only in terms of the frequency of items across the underlying transactions. Hence, the resulting notion of quality suffers from the typical limitations of the approaches, which use exact-match similarity measures to assess cluster homogeneity. To this purpose, conceptual cluster homogeneity for categorical data can be easily added to the framework of the EPF algorithm.

Another limitation of our approach is that it cannot deal with outliers. These are transactions whose structure strongly differs from that of the other transactions being characterized by low-frequency items. A cluster containing such transaction exhibits low quality. Worst, outliers could negatively affect the PARTITION-CLUSTER procedure by preventing the split to be accepted (because of an arbitrary assignment of such outliers, which would lower the quality of the partitions). Hence, a significant improvement of EPF can be obtained by defining an outlier detection procedure that is capable of detecting and removing outlier transactions before partitioning the clusters. Accordingly the research work is being focused further to improve the quality of clusters which are created after EPF.

REFERENCES

[1] J. Grabmeier and A. Rudolph, "Techniques of Cluster Algorithms in Data Mining," Data Mining and Knowledge Discovery, vol. 6, no. 4, pp. 303-360, 2002.

[2] A. Jain and R. Dubes, Algorithms for Clustering Data. Prentice Hall, 1988.

[3] R. Ng and J. Han, "CLARANS: A Method for Clustering Objects for Spatial Data Mining," IEEE Trans. Knowledge and Data Eng., vol. 14, no. 5, pp. 1003-1016, Sept./Oct. 2002.
[4] Z. Huang, "Extensions to the K-Means Algorithm for Clustering Large Data Sets with Categorical Values," Data Mining an Knowledge Discovery, vol. 2, no. 3, pp. 283-304, 1998.
[5] C. Gozzi, F. Giannotti, and G. Manco, "Clustering Transactional Data," Proc. Sixth European Conf. Principles and Practice of Knowledge Discovery in Databases (PKDD '02), pp. 175-187, 2002.
[6] S. Deerwester et al., "Indexing by Latent Semantic Analysis," J. Am. Soc. Information Science, vol. 41, no. 6, 1990.
[7] L. Parsons, E. Haque, and H. Liu, "Subspace Clustering for High-Dimensional Data: A Review," SIGKDD Explorations, vol. 6, no. 1, pp. 90-105, 2004.
[8] G. Gan and J. Wu, "Subspace Clustering for High Dimensional Categorical Data," SIGKDD Explorations, vol. 6, no. 2, pp. 87-94, 2004.
[9] M. Zaki and M. Peters, "CLICK: Mining Subspace Clusters in categorical Data via k-Partite Maximal Cliques," Proc. 21st Int'l Conf. Data Eng. (ICDE '05), 2005.
[10] Y. Yang, X. Guan, and J. You, "CLOPE: A Fast and Effective Clustering Algorithm for Transactional Data," Proc. Eighth ACM Conf. Knowledge Discovery and Data Mining (KDD '02), pp. 682-687, 2002.
[11] E. Han, G. Karypis, V. Kumar, and B. Mobasher, "Clustering in a High Dimensional Space Using Hypergraph Models," Proc. ACM SIGMOD Workshops Research Issues on Data Mining and Knowledge Discovery (DMKD '97), 1997.
[12] M. Ozdal and C. Aykanat, "Hypergraph Models and Algorithms for Data-Pattern-Based Clustering," Data Mining and Knowledge Discovery, vol. 9, pp. 29-57, 2004.
[13] K. Wang, C. Xu, and B. Liu, "Clustering Transactions Using Large Items," Proc. Eighth Int'l Conf. Information and Knowledge Management (CIKM '99), pp. 483-490, 1999.
[14] D. Barbara´ , J. Couto, and Y. Li, "COOLCAT: An Entropy-Based Algorithm for Categorical Clustering," Proc. 11th ACM Conf. Information and Knowledge Management (CIKM '02), pp. 582-589, 2002.
[15] P. Andritsos, P. Tsaparas, R. Miller, and K. Sevcik, "LIMBO: Scalable Clustering of Categorical Data," Proc. Ninth Int'l Conf. Extending Database Technology (EDBT '04), pp. 123-146, 2004.
[16] M.O.T. Li and S. Ma, "Entropy-Based Criterion in Categorical Clustering," Proc. 21st Int'l Conf. Machine Learning (ICML '04), pp. 68-75, 2004.
[17] I. Cadez, P. Smyth, and H. Mannila, "Probabilistic Modeling of Transaction Data with Applications to Profiling, Visualization, and Prediction," Proc. Seventh ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '01), pp. 37-46, 2001.
[18] M. Carreira-Perpinan and S. Renals, "Practical Identifiability of Finite Mixture of Multivariate Distributions," Neural Computation, vol. 12, no. 1, pp. 141-152, 2000.
[19] G. McLachlan and D. Peel, Finite Mixture Models. John Wiley & Sons, 2000.
[20] M. Meila and D. Heckerman, "An Experimental Comparison of Model-Based Clustering Methods," Machine Learning, vol. 42, no. 1/2, pp. 9-29, 2001.
[21] J.G.S. Zhong, "Generative Model-Based Document Clustering: A Comparative Study," Knowledge and Information Systems, vol. 8, no. 3, pp. 374-384, 2005.
[22] A. Gordon, Classification. Chapman and Hall/CRC Press, 1999.
[23] C. Fraley and A. Raftery, "How Many Clusters? Which Clustering Method? The Answer via Model-Based Cluster Analysis," The Computer J., vol. 41, no. 8, 1998.
[24] P. Smyth, "Model Selection for Probabilistic Clustering Using Cross-Validated Likelihood," Statistics and Computing, vol. 10, no. 1, pp. 63-72, 2000.
[25] D. Pelleg and A. Moore, "X-Means: Extending K-Means with Efficient Estimation of the Number of Clusters," Proc. 17th Int'l Conf. Machine Learning (ICML '00), pp. 727-734, 2000.
[26] M. Sultan et al., "Binary Tree-Structured Vector Quantization Approach to Clustering and Visualizing Microarray Data," Bioinformatics, vol. 18, 2002.
[27] S. Guha, R. Rastogi, and K. Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," Information Systems, vol. 25, no. 5, pp. 345-366, 2001.
[28] J. Basak and R. Krishnapuram, "Interpretable Hierarchical Clustering by Constructing an Unsupervised Decision Tree," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 1, Jan. 2005.
[29] H. Blockeel, L.D. Raedt, and J. Ramon, "Top-Down Induction of Clustering Trees," Proc. 15th Int'l Conf. Machine Learning (ICML'98), pp. 55-63, 1998.
[30] B. Liu, Y. Xia, and P. Yu, "Clustering through Decision Tree Construction," Proc. Ninth Int'l Conf. Information and Knowledge Management (CIKM '00), pp. 20-29, 2000.
[31] Yi-Dong Shen, Zhi-Yong Shen and Shi-Ming Zhang,"Cluster Cores – based Clustering for High – Dimensional Data".
[32] Alexander Hinneburg and Daniel A. Keim, Markus Wawryniuk,"HD-Eye-Visual of High-Dimensional Data: A Demonstration".
[33] http://en.wikipedia.org/wiki/Bayes'_theorem
[34] UCI Machine Learning Repository http://www.ics.uci.edu/~mlearn/
[35] D. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," Machine Learning, vol. 2, pp. 139-172, 1987.
[36] V. Ganti, J. Gehrke, and R. Ramakrishnan, "CACTUS: Clustering Categorical Data Using Summaries," Proc. Fifth ACM Conf. Knowledge Discovery and Data Mining (KDD '99), pp. 73-83, 1999.
[37] D. Gibson, J. Kleinberg, and P. Raghavan, "Clustering Categorical Data: An Approach Based on Dynamical Systems," VLDB J., vol. 8, pp. 222-236, 2000.

Prasad S.Halgaonkar received his B.E. in Computer Science and Engg. from Amravati University in 2006. Currently, he is pursuing his M.Tech (CSE) from Walchand College, Shivaji University. His current research interest includes Distributed Data Mining, Cognitive Radio and Wireless Communication.

Dr. Vijay M.Wadhai received his B.E. from Nagpur University in 1986, M.E. from Gulbarga University in 1995 and Ph.D. degree from Amravati University in 2007. He has experience of 24 years which includes both academic (17 years) and research (7 years). He has been working as a Dean of Research, MITSOT, MAE, Pune (from 2009) and simultaneously handling the post of Director - Research and Development, Intelligent Radio Frequency (IRF) Group, Pune (from 2009). His research interest includes Deductive Databases, Knowledge Discovery and Data Mining, Cognitive Radio and Wireless Communication, Spectrum Management, Wireless Sensor Network, ASIC Design - VLSI, Advance Network Design. He is a member of LMISTE, MIETE, MIEEE, MIES and GISFI (Member Convergence Group), India.

A.D.Potgantwar received B.E. (CSE) degree from Amravati University in 2005, M.Tech (CSE) from VJTI Mumbai, Mumbai University in 2009. After working as a lecturer (2005-07) in the D.N.Patel College of Engineering Shahada, North Maharashtra University, he has been a lecturer at Pune Univ. since 2009 and simultaneously handling the post of Director- Journal, Intelligent Radio Frequency (IRF) Group, Pune. His research interest includes Knowledge Discovery and Data Mining, Cognitive Radio and Wireless Communication, Image Processing, VHDL.