BILCOM: Bi-level Clustering of Mixed Categorical and Numerical Biological Data

Bill Andreopoulos

Aijun An

Xiaogang Wang

Technical Report CS-2005-01

January 2005

Department of Computer Science and Engineering

4700 Keele Street North York, Ontario M3J 1P3 Canada

BILCOM: Bi-Level Clustering of Mixed Categorical and Numerical Biological Data

Bill Andreopoulos[1], Aijun An[1], and Xiaogang Wang[2]

[1]Department of Computer Science and Engineering, York University, 4700 Keele Street, Toronto, M3J1P3, Ontario, Canada, {billa,aan}@cs.yorku.ca
[2]Department of Mathematics and Statistics, York University, 4700 Keele Street, Toronto, M3J1P3, Ontario, Canada, stevenw@mathstat.yorku.ca

Abstract. Data sets emerging from biomedical domains often have mixed categorical and numerical types, where the categorical type represents semantic information on the objects, while the numerical represents experimental results. We present the BILCOM algorithm for "Bi-Level Clustering of Mixed data types". BILCOM clusters data sets of objects that have numerical attribute values, while incorporating categorical attribute values. This clustering algorithm performs a pseudo-Bayesian process, where the prior is categorical clustering and the posterior is numerical clustering. This algorithm provides a different type of insight for biomedical data sets by giving their 'full picture'. We compare BILCOM with traditional clustering algorithms applied to biomedical data sets of mixed types, including yeast, hepatitis and thyroid disease. The results indicate that BILCOM partitions data sets of mixed types more accurately than if using one type alone.

1 Introduction

Large data sets emerging from studies in biomedical research are often analyzed using clustering tools. Clustering aims to partition a set of objects into groups, so that objects with similar characteristics are grouped together and different groups contain objects with dissimilar characteristics [10,14,16,18]. Often when clustering is applied to biomedical data sets of objects with numerical attribute values (NAs), the process does not incorporate the semantic information that has been deposited in databases as categorical attribute values (CAs) on the objects [7,12,16,23]. We present the BILCOM algorithm for clustering data sets of objects with mixed CAs and NAs. This algorithm clusters numerical data, incorporating semantic information in the form of CAs. Characteristics of this clustering approach include: *a.* If little categorical similarity can be found between an object and a cluster, then the object will be clustered instead based on numerical similarity, thus increasing its chance to be clustered correctly. *b.* BILCOM clustering is not based on local decisions and there is an opportunity to re-evaluate the clusters later in the process [10,14,16,18]. *c.* BILCOM clustering uses CAs during the clustering process, unlike other techniques that annotate the clusters with CAs after the process [19].

Data sets for which BILCOM clustering is particularly useful exist in the domain of evidence-based medicine. In these data sets the CAs represent the characteristics or symptoms of patients and NAs represent the results of medical experiments on patients. BILCOM applied to clustering such medical data sets can produce clusters reflecting the medical outcome of patients. Another important application area for BILCOM are microarray gene expression data sets that contain CAs representing known gene functions [7,12,23] and NAs representing gene expression across time or across tissues [8,9,26].

This paper is organized as follows. Section 2 describes previous related work. Section 3 describes the BILCOM clustering algorithm. Section 4 presents the pseudo-Bayesian rationale for the BILCOM algorithm. Section 5 describes application to real yeast data sets. Section 6 discusses experimental results for applying BILCOM to hepatitis and thyroid disease patient data sets. Section 7 discusses selecting the appropriate BILCOM parameter values. Finally, Section 8 concludes the paper.

2 Background on Clustering Algorithms for Mixed Data Types

Several algorithms have been proposed for clustering mixed categorical (discrete) and numerical (discrete or continuous) data types. An object o has m attributes $\{o_1,...,o_m\}$. Each o_i, $i=1..m$, has a value taken from a domain $S=\{s_1,...,s_x\}$ whose values may be of categorical or numerical data types. A domain of categorical data type is SEX with the values M or F that have no ordering defined. A domain of numerical data type is GPA with ordered values in the range 0.0-4.0. In this paper m represents the number of categorical attributes in each object and N represents the number of objects in the data set.

AutoClass can cluster mixed categorical and numerical data based on prior distributions [27]. It does not require the user to specify the number of clusters beforehand. AutoClass uses a Bayesian method for determining the optimal classes. AutoClass takes a prior distribution of each attribute in each cluster, symbolizing the prior beliefs of the user. It changes the classifications of items in clusters and changes the means and variances of the distributions, until the means and variances stabilize.

k-Modes is a clustering algorithm that deals with categorical data [20,21]. The k-Modes clustering algorithm requires the user to specify the number of clusters to be produced and the algorithm builds and refines the specified number of clusters. Each cluster has a mode associated with it. Assuming that the objects in the data set are described by m categorical attributes, the mode of a cluster is a vector $Q=\{q_1, q_2, \ldots, q_m\}$ where q_i is the most frequent value for the ith attribute in the cluster of objects. A similarity metric is needed to choose the closest cluster to an object by computing the similarity between the cluster's mode and the object. Let $X=\{x_1, x_2, \ldots, x_m\}$ be an object, where x_i is the value for the ith attribute. The similarity between X and Q is defined as:

$$\text{similarity}(X, Q) = \sum_{i=1}^{m} \sigma(x_i, q_i) \qquad \sigma(x_i, q_i) = \begin{cases} 1 & (x_i = q_i); \\ 0 & (x_i \neq q_i). \end{cases}$$

An extension of k-Modes called *k-Prototypes* was proposed in [22] to deal with mixed numerical and categorical data. K-Prototypes also adopts an iterative approach to clustering that continues until objects stop changing clusters.

ROCK is a hierarchical clustering algorithm for categorical data [17]. ROCK assumes a similarity measure between tuples and defines a link between two tuples whose similarity exceeds a threshold w. Initially, each tuple is assigned to a separate cluster and then clusters are merged repeatedly according to the closeness between clusters. The closeness between clusters is defined as the sum of the number of "links" between all pairs of tuples, where the number of "links" represents the number of common neighbors between two clusters.

Supervised learning and *Support Vector Machines* classify objects based on prior knowledge. Supervised learning draws a boundary separating classes, based on a training data set of labeled objects. Future unlabelled objects are classified on one side of the boundary. Some applications of SVMs to biomedical domains can be found in [15,26].

3 The BILCOM Clustering Algorithm

The *BILCOM "Bi-Level Clustering of Mixed categorical and numerical data types*" algorithm performs clustering at two levels, where the first level clustering acts as a prior for the second level, thus simulating a pseudo-Bayesian process as described later in Section 4. The data sets come primarily from the biomedical domain. In these sets, categorical attribute values (CAs) represent semantic information on the objects, while numerical attribute values (NAs) represent experimental results. By Bayesian theory it makes sense to use CAs at the first level and numerical attributes at the second level, rather than start by using numerical data and then categorical. Similarity based on CAs is emphasized at the first level and similarity based on NAs at the second level. The first level result is the prior that is given as input to the second level and the second level result is the output of BILCOM. Figure 1 shows an example, where the first level and second level involve four clusters. The second level clusters consist of subclusters. Object A is assigned to different first and second level clusters, because the NA similarity at the second level is stronger than the CA similarity at the first level. The following relationship holds for A:

$$\begin{aligned} &\textit{categorical\_similarity}(A, \textit{cluster}2) + \\ &\textit{numerical\_similarity}(A, \textit{cluster}2) > \\ &\textit{categorical\_similarity}(A, \textit{cluster}3) + \\ &\textit{numerical\_similarity}(A, \textit{cluster}3) \end{aligned}$$

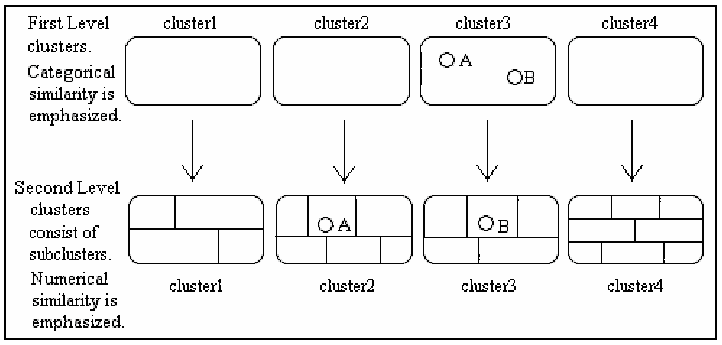

Fig. 1. Overview of the BILCOM clustering process.

On the other hand, object B is assigned to the same clusters in both levels, because both CA and NA similarities support this classification. Thus, BILCOM considers CA and NA similarities of an object to the clusters to which it may be assigned.

Different types of data are used at the first and second levels. The numerical data represent experimental results involving the objects. For example, the numerical data used at the second level might look as follows: BILIRUBIN:0.39; ALBUMIN:2.1; PROTIME:10. The categorical data represent what was observed to be true about the objects before the experiment. For example, the categorical data used at the first level might be existing information on objects looking as follows: SEX:male; STEROID:yes; FATIGUE:no; ANOREXIA:no. BILCOM clustering has the following characteristics:

a. The coherence of each cluster is maximized, considering both numerical and categorical similarity of the objects.

b. Only the objects with highest categorical similarity to a cluster form the basis for clustering at the first level.

c. The results of the first level clustering, which is the prior for the process, do not exert an overly strong effect on the second level, so that the second level clustering can escape a poor prior.
d. The number of clusters to be formed does not need to be specified by the user beforehand.

3.1 Design of BILCOM

This section describes the first level and second level algorithms that form the BILCOM process, shown in Figure 1. The first level is the MULIC categorical clustering algorithm [2] and it clusters only a subset of the data set objects. The reason MULIC was chosen for the first level is that it creates multiple layers for each cluster and objects in top layers are more likely to be clustered correctly than objects in bottom layers. Thus, it is easy to select the objects in the top layers of MULIC clusters, as the most reliable classifications for the first level of BILCOM.

3.1.1 First level clustering

At the first level, clustering is performed using MULIC [2], an extension of the k-Modes clustering algorithm for categorical data sets that was discussed in Section 2 [20,21]. The k-Modes clustering algorithm requires the user to specify the number of clusters to be produced and the algorithm builds and refines the clusters. Each cluster has a mode associated with it. Assuming that the objects in the data set are described by m categorical attributes, the mode of a cluster is a vector $Q=\{q_1, q_2, \ldots, q_m\}$ where q_i is the most frequent value for the ith attribute in the given cluster.

The MULIC clustering algorithm makes substantial changes to k-Modes. MULIC ensures that when each object is clustered it is inserted into the cluster with the most similar mode, thus maximizing the similarity between the object and the mode:

$$similarity(o_i, \text{mode}_i) \qquad (1)$$

where o_i is the ith object in the data set and $mode_i$ is the mode of the ith object's cluster. The similarity metric is the k-Modes similarity, described in Section 2, which returns the number of identical CAs between an object and a mode.

The MULIC algorithm has the following characteristics. First, the number of clusters is not specified by the user. Clusters are created, removed or merged during the clustering process, as the need arises. Second, it is possible for all objects to be assigned to clusters of size two or greater by the end of the process. Third, clusters are layered.

Figure 2 shows the main part of the MULIC clustering algorithm. The algorithm starts by reading all objects from the input file and storing them in S. The first object is inserted in a new cluster, the object becomes the mode of the cluster and the object is removed from S. Then, it continues iterating over all objects that have not been assigned to clusters yet, to find the closest cluster. In all iterations, the closest cluster for each unclassified object is the cluster with the highest similarity between the cluster's mode and the object, as computed by the similarity metric [20,21].

The variable φ is maintained to indicate how strong the similarity has to be between an object and the closest cluster's mode for the object to be inserted in the cluster – initially φ equals 0, meaning that the similarity has to be very strong between an object and the closest cluster's mode. If the number of different CAs between the object and the closest cluster's mode is greater than φ then the object is inserted in a new cluster on its own, else, the object is inserted in the closest cluster and the mode is updated.

At the end of each iteration, all objects classified in clusters of size one have their clusters removed so that the objects will be re-clustered at the next iteration. This ensures that the clusters that persist through the process are only those containing at least 2 objects for which the required similarity can be found. Objects belonging to clusters with size greater than one are removed from the set of unclassified objects S, so those objects will not be re-clustered.

At the end of each iteration, if no objects have been inserted in clusters of size greater than one, then the variable φ is incremented by $\delta\varphi$. Thus, at the next iteration the criterion for inserting objects in clusters will be more flexible. The iterative process stops when all objects are classified in clusters of size greater than one, or φ exceeds a user-specified *threshold*. If the threshold equals its default value of the number of attributes m, the process stops when all objects are assigned to clusters of size greater than one.

The MULIC algorithm can eventually classify all objects in clusters, even if the closest cluster to an object is not that similar, because φ can continue increasing until all objects are classified. Even in the extreme cases, where an object o with m attributes has only zero or one value similar to the mode of the closest cluster, it can still be classified when $\varphi = m$ *or* $\varphi = m-1$.

Input: (1) a set *S* of objects;
Parameters: (1) $\delta\varphi$: the increment for φ;
(2) *threshold* for φ : the maximum number of values that can differ between an object and the mode of its cluster;
Default parameter values: (1) $\delta\varphi = 1$;
(2) *threshold* = the number of categorical attributes *m*;
Output: a set of clusters;
Method:

1. Insert the first object into a new cluster, use the object as the mode of the cluster, and remove the object from *S*;
2. Initialize φ to 0;
3. Loop through the following until S is empty or φ is greater than the specified *threshold*
   a. For each object *o* in *S*
      i. Find *o*'s closest cluster *c* by using the similarity metric to compare *o* with the modes of all existing cluster(s);
      ii. If the number of different values between *o* and *c's* mode is larger than φ, insert o into a new cluster
      iii. Otherwise, insert *o* into *c* and update *c's mode;*
      iv. Remove object *o* from *S*;
   b. For each cluster *c*, if there is only one object in *c*, remove *c* and put the object back in *S;*
   c. If in this iteration no objects were inserted in a cluster with size > 1, increment φ by $\delta\varphi$.

Fig. 2. The MULIC clustering algorithm.

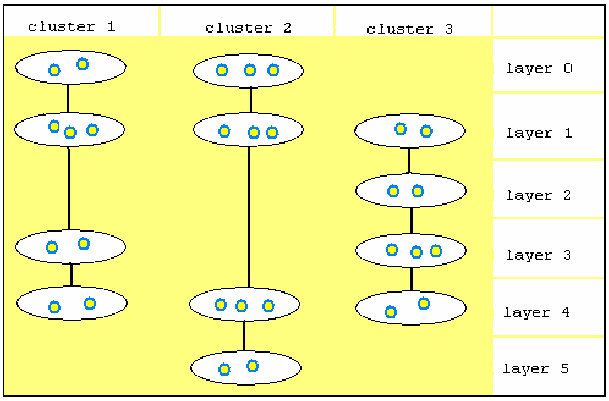

Fig. 3. MULIC results. Each cluster consists of one or more different layers representing different similarities of the objects attached to the cluster.

Figure 3 illustrates what the results of MULIC look like. Each cluster consists of many different "layers" of objects. The layer of an object represents how strong the object's similarity was to the mode of the cluster when the object was assigned to the cluster. The cluster's layer in which an object is inserted depends on the value of φ. Lower layers have a lower coherence - meaning a lower average similarity between all pairs of objects in the layer - and correspond to higher values of φ. MULIC starts by inserting as many objects as possible in high layers – such as layer 0 or 1 - and then moves to lower layers, creating them as φ increases.

If an unclassified object has equal similarity to the modes of the two or more closest clusters, then the algorithm tries to resolve this 'tie' by comparing the object to the mode of the top layer of each of these clusters – the top layer of a cluster may be layer 0 or 1 or 2 and so on. Each cluster's top layer's mode was stored by MULIC when the cluster was created, so it does not need to be recomputed. If the object has equal similarity to the modes of the top layer of all of its closest clusters, the object is assigned to the cluster with the highest bottom layer. If all clusters have the same bottom layer then the object is assigned to the first cluster, since there is insufficient data for selecting the best cluster.

The complexity of MULIC is $O(N^2)$, where N is the number of objects. Most of our trials had a runtime of less than 1 second. Increasing $\delta\varphi$ or decreasing *threshold* reduces the runtime, often without hurting the quality of results [2].

The question remains of which objects to be clustered at the first level. The first level objects are those whose comparison to the mode of the closest cluster by the similarity metric yields a result that is greater than or equal to a value *minimum_mode_similarity*, while the rest of the objects are clustered at the second level. The user can specify a value for the *threshold* for φ that is less than its default value of the number of categorical attributes *m*. This *threshold* value for φ is *m-minimum_mode_similarity*. When φ exceeds the maximum allowed value specified by *threshold*, any remaining objects are clustered at the second level instead. The reason only the objects whose similarity to the closest mode is greater than *minimum_mode_similarity* are clustered at the first level is because the objects that yield a low similarity to the closest mode are more likely to be inserted in a wrong cluster, as we showed in [1,2]. Thus, the objects whose classification in clusters based on categorical similarity is not reliable enough are clustered at the second level instead, where the numerical similarity of objects to clusters is more influential. We discuss setting the values of *threshold* and *minimum_mode_similarity* in Sect. 6.

3.1.2 Second level clustering

The first level result is the input to the second level. The second level clusters all of the data set objects, including the objects clustered at the first level. The second level uses numerical data type similarity and the first level result as a prior. The second level clustering consists of 5 steps, whose rationale is to simulate maximizing the numerator of the Bayesian equation, as described in [4]. The second level result is the output of the BILCOM process.

Step 1. One object in each first level cluster is set as a *seed*, while all the rest of the objects in the cluster are set as *centers*. The seed is an object that is at the top layer of the cluster – ideally in layer 0. The reason we choose for seed a top layer object is that the most influential objects at the second level should be those that have the minimum average distance to all other objects in the first level cluster. The MULIC paper [2] showed that objects at the top layer have a smaller average distance to all other cluster objects than lower layer objects do.

If the top layer of a cluster is layer 0 then we have no difficulty in choosing the seed since all objects have the same CAs. If the top layer of a cluster is not layer 0 and it contains more than one object, then we choose the seed by comparing all top layer objects to the cluster's mode to find the closest object. If this does not resolve the ambiguity then we compare all top layer objects to the cluster's top layer mode – which was stored by MULIC when the cluster was created - to find the closest object. If all top layer objects have the same similarities to modes then we assign the seed to be the first top layer object, since there is insufficient information for choosing the best seed.

Step 2. Each *seed* and *center* is inserted in a new *second level subcluster*. The output of this step is a set of *subclusters*, referred to as *seed-containing* or *center-containing* subclusters, whose number equals the number of objects clustered at the first level.

Step 3. Each object that did not participate at the first level is inserted into the second level subcluster containing the most numerically similar *seed* or *center*. Numerical similarity for Steps 3-5 is determined by the *Pearson correlation coefficient* or the *Shrinkage-based similarity metric* introduced by Cherepinsky et al [6].

Step 4. Each center-containing subcluster is merged with its most numerically similar seed-containing subcluster. The most numerically similar seed-containing subcluster is found using our version of the ROCK goodness measure [17] that is evaluated between the center-containing subcluster in question and all seed-containing subclusters:

$$G(C_i, C_j) = \frac{link[C_i, C_j]}{size(C_i) \times size(C_j)}$$

$link[C_i,C_j]$ stores the number of cross links between subclusters C_i and C_j, by evaluating $\Sigma(o_q \in C_i,\ o_r \in C_j)\ link(o_q,o_r)$. $link(o_q,o_r)$ is a boolean value specifying whether a link exists between objects o_q and o_r . A link is set between two objects if the objects' numerical similarity is higher than a value *minimum_numerical_similarity*. The rationale for using a variation of ROCK's goodness measure for this step is that the link-based approach of ROCK adopts a global approach to the clustering problem, by capturing the global information about neighboring objects between clusters. It has been shown to be more robust than methods that adopt a local approach to clustering, like hierarchical clustering [17].

Step 5. The loop below refines the step 4 subcluster merges. All variables take real values in the range 0.0-1.0.

```
repeat {
    foreach (center-containing_subcluster)
       if (numerical_similarity_of_center_subcluster_to_1st_level_seed_cluster ×
categorical_similarity_of_center_to_seed_of_1st_level_cluster >
numerical_similarity_of_center_subcluster_to_its_numerically_similar_2nd_level_cluster ×
categorical_similarity_of_center_to_seed_of_its_numerically_similar_2nd_level_cluster)
             merge center-containing_subcluster to seed-containing_subcluster from 1st_level;
} until (no center-containing_subcluster changes);
```

The variable:

```
categorical_similarity_of_center_to_seed_of_1st_level_cluster
```

represents the *categorical* similarity of the center *c* of a subcluster *C* to the seed *s*, such that *c* and *s* were in the same first level cluster.

The variable:

```
categorical_similarity_of_center_to_seed_of_its_numerically_similar_2nd_level_cluster
```

represents the *categorical* similarity of the center *c* of a subcluster *C* to the seed of C's most numerically similar seed-containing subcluster *N* determined in step 4. The categorical similarity is computed as follows:

$$\text{similarity}(\text{center},\text{seed}) = \frac{\sum_{i=1}^{m} \sigma(\text{center}_i, \text{seed}_i)}{m} \qquad \sigma(\text{center}_i,\text{seed}_i)= 1 \text{ if } \text{center}_i=\text{seed}_i \text{ , 0 otherwise.}$$

The variables:

numerical_similarity_of_center_subcluster_to_1st_level_seed_cluster
numerical_similarity_of_center_subcluster_to_its_numerically_similar_2nd_level_cluster

represent the *numerical* similarity of a subcluster *C* containing center *c* to the cluster containing seed *s*, such that *c* and *s* were in the same first level cluster, and to the cluster containing C's most numerically similar seed-containing subcluster *N* determined in step 4, respectively. These similarities include the subclusters that were merged to the clusters in previous iterations of the loop.

According to this loop, a subcluster *C* containing *center c* is attracted to the subcluster *S* containing *seed s*, such that *c* and *s* were in the same first level cluster. The attraction is stronger if there is high categorical similarity between *c* and *s* and lower if there is low categorical similarity between *c* and *s*. The subclusters *C* and *S* get merged if both the categorical similarity between *c* and *s* and numerical similarity between *C* and *S* are high enough. If *c* is not categorically similar enough to *s*, then, *C* should be likely to remain merged with its most numerically similar seed-containing subcluster *N* determined in step 4. Figure 4 shows steps 4 and 5.

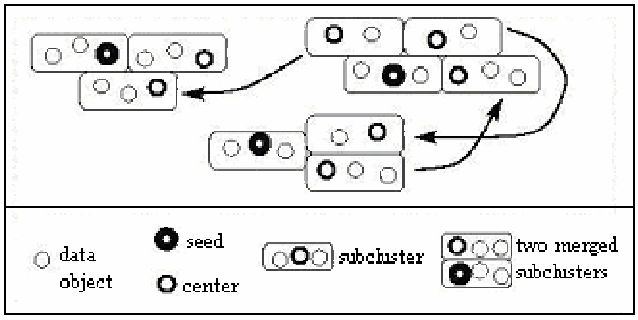

Fig. 4. Steps 4 and 5 of the second level of BILCOM clustering.

In Section 6.2, below Table 12, we discuss tests to show that BILCOM is able to *escape a poor prior* – for instance, if a center *c* was inserted in a first level cluster with weak similarity to the cluster mode, or if the similarity to the mode was erroneously high enough, or if *c* had wrong CAs with low confidence to be correct. The categorical similarity between the center *c* and the seed *s*, such that *c* and *s* were in the same first level cluster, is likely to return a low value when the prior is poor. In this case, the subcluster *C* containing center *c* will be likely to remain merged with its most numerically similar seed-containing subcluster *N* determined in step 4, instead of the subcluster *S* containing the seed *s*. Thus, the prior can be escaped and the data can be clustered correctly. In this case, *C* will not be merged to *S*, unless their numerical similarity is very high.

On the other hand, if the subcluster *C* containing center *c* is merged to the subcluster *S* containing the seed *s*, such that *c* and *s* were in the same first level cluster, then *C* must be numerically similar enough to *S*. This way we ensure that if a subcluster *C* is merged to the subcluster *S* that is suggested by the results of the first level clustering, the numerical similarity between *C* and *S* is high enough to support the merging.

The reason why the inequality comparison in step 5 considers the seeds of clusters, instead of the cluster modes, is that by considering similarity to seeds we are effectively giving the objects a *second chance* to reorganize and to escape their first level clustering, if the first level clustering was weak. Since the first level clustering was based on comparisons to modes that often yield wrong results and, therefore, objects may be attached to wrong clusters, the comparison in step 5 allows the similarities to be reconsidered. We showed in [2] that objects in the top layers 0 and 1 such as seeds have a higher average similarity to all other cluster objects than do lower layer objects.

4 The Doctris Learning Framework

We introduce a framework providing the theoretical rationale for BILCOM, which we call *Doctris "Double Criteria Triple Step"*. Doctris assumes that two different criteria exist and will be used at two different levels of the clustering process, such that the result of the first level provides a prior for the second level. The *criteria* are objects' attributes with values of different domains, which could be of different data types such as numerical, categorical or boolean. The Doctris framework is built upon the Bayesian framework but is pseudo-Bayesian at this moment. As in the Bayesian framework, the probabilities estimated at the first level are updated or corrected at the second level. Therefore, it bears a great similarity with the empirical

Bayes method. However, the probabilities are estimated by the similarity metric instead of using the likelihood as in the true Bayesian paradigm. The framework is based on the idea that a high similarity between an object and a cluster means a high probability that this is the correct cluster for the object, while a lower similarity means a lower probability. This approach lays a path to incremental empirical learning with more than one criterion, thus tackling an important problem.

1. Two criteria A and B exist. For example, similarity between objects could be computed on the basis of numerical and categorical data type values.
2. The input is a set of N objects $D = \{D_1...D_N\}$.
3. The output is a set of X classes $C = \{C_1...C_X\}$ that are learnt from D, such that $X \leq N$.
4. The process of learning C from D consists of 3 steps:

First level:

Step 1. On the basis of criterion A, form a number of X classes $C = \{C_1...C_X\}$, that are most likely to have as few misclassified objects as possible. Only M of the objects in D are classified, where $X \leq M \leq N$. Using more objects than M would increase the likelihood of misclassified objects in C.

Second level:

Step 2. On the basis of criterion B, form a number of M subclasses $d = \{d_1...d_M\}$, by matching each object in the set $u=\{u_1...u_{N-M}\}$ of N-M objects not classified in Step 1, to the closest one of the M objects classified in Step 1. Each subclass d_i, $i=1...M$, consists of one of the M objects classified in Step 1 and zero or more of the objects in u not classified in Step 1.

Step 3. On the basis of both criteria A and B, selectively merge the M subclasses $d = \{d_1...d_M\}$ to one another, until a set of X classes $C = \{C_1...C_X\}$ emerges.

Fig. 5. The Doctris framework for unsupervised learning with two criteria.

Figure 5 presents the general steps that define the Doctris framework. As shown, classification is performed on the basis of two criteria A and B. The first level consists of Step 1 that is based on criterion A. The second level consists of Steps 2 and 3 that are based on criteria A and B. When learning in Step 1 on the basis of criterion A, the likelihood of misclassification will increase to an unacceptable level after a number of M objects have been classified. A clustering algorithm for which this assumption holds is the MULIC clustering algorithm [2]. After M objects have been classified in Step 1, the unclassified N-M objects are classified in Steps 2 and 3 based on both of the criteria A and B. In Step 2, each one of the remaining N-M objects is matched to the closest of the M objects based on criterion B, thus resulting in M subclasses. In Step 3, the subclasses resulting from Step 2 are selectively merged to one another, based on both criteria A and B, until X classes emerge from the process. This sequence of steps simulates a Bayesian process described in Section 4.1, where the result of Step 1 is the prior for Steps 2 and 3.

The general steps that define the Doctris framework serve the ultimate purpose of enhancing the classification process to produce more accurate results. The process will be significantly improved if certain rules are satisfied. In the descriptions that follow $lmcAandB[M_A]$ is the "likelihood of misclassification of objects based on criteria A and B, after M objects have been classified based on criterion A". $lmcA[M_A]$ is the "likelihood of misclassification of objects based on criterion A, after M objects have been classified based on criterion A".

In order for a Doctris algorithm to produce more accurate results than Step 1 of the framework would produce alone based on criterion A only, the following inequality needs to be satisfied:

$$lmcAandB[M_A] \times \frac{(N-M)+(M-X)}{N} < lmcA[M_A] \times \frac{N-M}{N}$$
$$\Leftrightarrow lmcAandB[M_A] \times (N-X) < lmcA[M_A] \times (N-M)$$
$$\Leftrightarrow lmcAandB[M_A] < lmcA[M_A] \times \frac{(N-M)}{(N-X)} \qquad (1)$$

Inequality (1) states that from the total number of objects $(N-M)+(M-X)=N-X$ whose classification may change in Steps 2 and 3 based on criteria A and B, fewer objects should be likely to be misclassified, than if using only criterion A in Step 1 to classify the remaining N-M objects. $(N-M)$ is the number of objects that are matched to a subclass in Step 2, while $(M-X)$ is the number of merges between subclasses that may occur in Step 3. The products estimate the number of objects that are likely to be misclassified, based on the likelihoods of misclassification.

As M increases, the term $lmcAandB[M_A]$ on the left-hand side of (1) remains stable because we are assuming both criteria A and B will be used and any objects misclassified in Step 1 will be given a second chance to be classified correctly during Steps 2 and 3. We would like to estimate the value of M such that the left-hand side of (1) is lower than the right-hand side of

(1) and the difference between the left-hand side and the right-hand side is maximized. We would like to use the value of M such that the following ratio is maximized:

$$\frac{lmcA[M_A] \times (N-M)/(N-X)}{lmcAandB[M_A]}$$

As M increases the term $lmcA[M_A]$ increases, at constant or varying rates for different values of M. However, as M increases the ratio $(N\text{-}M)/(N\text{-}X)$ decreases at a constant rate since N and X are constants; since $X \leq M \leq N$, this ratio is in the range 0.0 to 1.0. A maximal product of the terms on the right-hand side of (1) can be estimated fairly easily. For example, 0.9×0.1 and 0.1×0.9 return 0.09, but 0.5×0.5 returns 0.25. A conservative approach would be to classify few objects in Step 1 such that $(N\text{-}M)/(N\text{-}X)$ is high while $lmcA[M_A]$ remains relatively low. A bolder approach would be to classify many objects in Step 1 such that $(N\text{-}M)/(N\text{-}X)$ is low while $lmcA[M_A]$ increases as much as possible.

Figures 6-7 show the graphs for two cases of the products of $lmcA[M_A]$ and $(N\text{-}M)/(N\text{-}X)$, for $X=10$ and $N=100$. M is represented by the horizontal axis. The term $(N\text{-}M)/(N\text{-}X)$ has a constant decrease rate. In the first case, $lmcA[M_A]$ increases at a rapid rate with M. The maximal product value of the two terms is at $M=35$. In the second case, $lmcA[M_A]$ increases at a lower rate with M. The maximal product value of the two terms is at $M=25$.

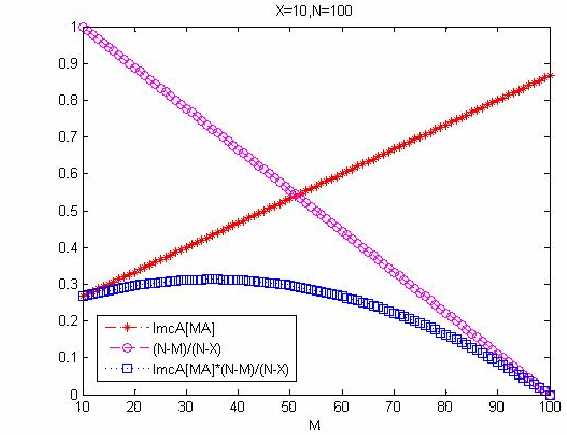

Fig. 6. This graph shows how $lmcA[M_A]$ increases at a *high* rate, while $(N\text{-}M)/(N\text{-}X)$ decreases at a constant rate. Number of clusters X is 10 and number of objects N is 100. M ranges between X and N as represented by the horizontal axis. The product value is maximized at an M value of 35.

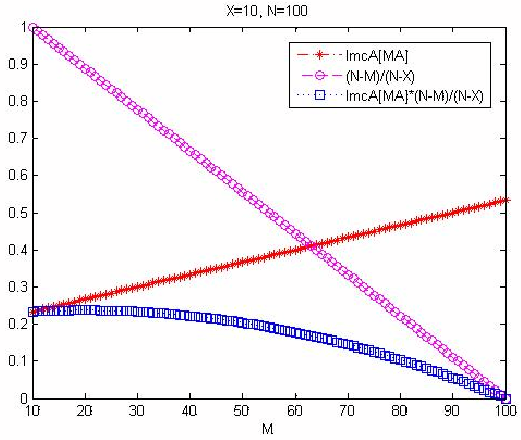

Fig. 7. This graph shows how $lmcA[M_A]$ increases at a *lower* rate, while $(N\text{-}M)/(N\text{-}X)$ decreases at a constant rate. Number of clusters X is 10 and number of objects N is 100. M ranges between X and N as represented by the horizontal axis. The product value is maximized at an M value of 25.

4.1 Pseudo-Bayesian Rationale for the Doctris Framework

Doctris is based on the Bayesian theory of classification. The idea of Bayesian classification is to find for each data object, the classification H for which the following Bayesian equation gives the maximum result:

$$\pi(H \mid E) = \frac{\pi(E \mid H)\pi(H)}{\pi(E)} \qquad (2)$$

$\pi(H)$ is the prior probability of hypothesis H. $\pi(E)$ is the prior probability that some evidence E is observed on the data object x. $\pi(E|H)$ is the likelihood of E given H. $\pi(H|E)$ is the posterior probability of H given E. We seek the hypothesis H such that the numerator $\pi(E|H)\pi(H)$ of the Bayesian equation is maximized. In other words, we seek the classification H of each object for which this numerator returns the maximum value. This gives the most probable classification H for an object.

Linking the Doctris framework to Bayesian theory of classification, the prior E for an object is the classification of the object in a class in Step 1, based on criterion A. The hypothesis H for an object is the classification of the object in a class in Steps 2 and 3, based on criteria A and B. For example, the criterion A could be attributes of a categorical domain that represent information about the object or our observations before an experiment takes place. The criterion B could be attributes of a numerical domain that represent the results of an experiment.

In this framework, $\pi(E)$ is a constant for each object that depends on the likelihood that the classification E of an object in a

class in Step 1 is correct. This likelihood increases with the strength of the similarity of the object to the class to which it is assigned in Step 1 according to criterion *A*. Since $\pi(E)$ is a constant for each object and it does not affect the choice between classifications *H* we do not use $\pi(E)$ in our calculations. In traditional Bayesian clustering algorithms, such as AutoClass, the Evidence *E* used in the Bayesian equation is given by the categorical or numerical attribute values of each object. In such traditional Bayesian clustering algorithms, the denominator $\pi(E)$ is often not considered in the process of finding the best classification. The term $\pi(H)$ is often not considered either, leaving the most important term to be $\pi(E|H)$.

In the following descriptions the term $similarity_{AB}(o,classStepx_o)$ is the similarity of object *o* to the class in which *o* is classified in Step *x*, according to the criteria *A* and *B*. This similarity can be computed using various algorithms, such as ROCK's similarity metric for numerical or categorical attribute value types [14].

Linking the Doctris framework to the Bayesian theory of classification, $\pi(H)$ is the likelihood that the classification of an object in a class in Steps 2 and 3 is correct – meaning that *H* is likely to be true. $\pi(H)$ increases with the strength of the similarity of the object to the subclass to which it is assigned in Step 2 according to criterion *B*. Since in Step 3 the subclasses are selectively merged to one another to form classes, to find $\pi(H)$ we are interested both in the similarity of the object to its Step 2 subclass according to criterion *B*, as well as the similarity of that subclass to the subclass to which it gets merged in Step 3 according to criteria *A* and *B*. Thus, we simulate maximizing $\pi(H)$ by seeking the classification *H* for an object *o* such that the following term is maximized:

$$\max(\pi(\mathrm{H})) =$$

$$\max(similarity_{AB}(o, classStep3_o)) =$$

$$\max(similarity_{\mathrm{B}}(o, subclassStep2_o) \times similarity_{\mathrm{AB}}(subclassStep2_o, classStep3_o)) \qquad (3)$$

Maximizing $\pi(E|H)$ assists our understanding of (3) above. Linking the Doctris framework to the Bayesian theory of classification, $\pi(E|H)$ is the likelihood that the classification of an object in a class in Step 1 is correct, meaning that *E* is likely to be true, given that the object is classified in Steps 2 and 3 in the class represented by *H*. $\pi(E|H)$ increases with the strength of the similarity of the object to the class to which it was assigned in Step 1 according to criterion *A*, as represented by *E* and $\pi(E)$. $\pi(E|H)$ also increases if the Step 1 class for the object, represented by *E,* is the same one as the class to which the object is assigned in Steps 2 and 3 based on criteria *A* and *B*, represented by *H*. It is obvious that maximizing $\pi(E|H)$ affects (3) above, in the sense that the $classStep3_o$ that previously yielded the highest similarity for *o* according to criteria *A* and *B* might not necessarily be the best choice. Instead, *o* might need to be assigned back to the class in which *o* was classified in Step 1, if the similarity of *o* to this class according to criteria *A* and *B* suggests this is a better choice. In our final decision on which class to assign *o* to, the similarity according to criteria *A* and *B* needs to be computed between *o* and the classes to which it was assigned in Steps 1 and 3, since both criteria are likely to contain information about the classification of object *o*. Thus, we simulate maximizing $\pi(E|H)$ by selecting between the Step 1 or Step 3 classification *H* for an object *o*:

$$\max(\pi(E \mid H)) =$$

$$\max(similarity_{AB}(o, classFinal_o)) =$$

$$\max(\max(\pi(H)), similarity_{AB}(o, classStep1_o)) =$$

$$\max(\max(similarity_{AB}(o, classStep3_o)), similarity_{AB}(o, classStep1_o)) \qquad (4)$$

Objects with low similarity to the closest Step 1 class according to criterion *A* are not classified in Step 1, thus ignoring for these objects the term $similarity_{AB}(o,classStep1_o)$. Such an object *o* will produce a higher value for $max(similarity_{AB}(o,classStep3_o))$ than $similarity_{AB}(o,classStep1_o)$ from (4), since Steps 2 and 3 will classify *o* based on both criteria *A* and *B*. Step 2 will assign *o* to the closest subclass based on criterion *B* and Step 3 will merge this subclass to the closest class based on criteria *A* and *B*, thus simulating maximizing $\pi(H)$ and $similarity_{AB}(o,classStep3_o)$. Furthermore, not classifying these objects in Step 1 reduces the total computation time.

The above similarity terms are defined below, where $metric_Z(x,y)$ represents a metric based on criterion *Z* for estimating the similarity between objects *x* and *y*, returning a value in the range 0.0 to 1.0 for low and high similarity between *x* and *y* respectively. For example, this metric could be the Euclidean distance for numerical attributes, or the modes-based similarity for categorical attributes as defined by Huang et al. [8,9]:

$\mathrm{similarity_B}(\mathrm{o,subclassStep2_o})$ can be estimated using $metric_B(o,y)$ where *y* is a representative object for $subclassStep2_o$.

$\mathrm{similarity_{AB}}(\mathrm{o,classStep1_o})$ can be estimated using $\alpha \times metric_A(o,y)+\beta \times metric_B(o,y)$, where *y* is a representative object for $classStep1_o$ and α and β are weights in the range 0.0 to 1.0, such that $\alpha+\beta=1.0$.

$\mathrm{similarity_{AB}}(\mathrm{subclassStep2_o,classStep3_o})$ can be estimated using:

$$\frac{\sum_{x \in subclassStep2_o, y \in classStep3_o} \alpha \times metric_A(x,y) + \beta \times metric_B(x,y)}{size(subclassStep2_o) \times size(classStep3_o)}$$

A Doctris algorithm takes into consideration all of these probabilities. The purpose is to classify each object in the class represented by H that maximizes the probability for the numerator $\pi(E|H)\pi(H)$ of the Bayesian equation.

4.2 An Evaluation Metric

The similarity metric of the Doctris framework can be adopted as a metric for evaluating the quality of the results. This would involve using $similarity_{AB}(o, classFinal_o)$ from (4) to calculate the average similarity of all objects o in the data set D to their respective classes. This evaluation metric can be described as follows:

$$Quality = \sum_{o \in D} similarity_{AB}(o, classFinal_o) / size(D) \qquad (5)$$

4.3 A Possible Extension of the Doctris Framework

The Doctris framework can be extended to more than two levels and criteria. New objects with a criterion Z may be presented to the classification process, after object classification has been done using criteria $A \ldots Y$. Criterion Z's attribute values might be of the same or different domains as the previous criteria $A \ldots Y$. For example, Z might be numerical while the previous criteria were categorical. In either case, criterion Z is presented to the classification process at a different time point from criteria $A \ldots Y$. The new objects possess the previous criteria $A \ldots Y$ as well as the new criterion Z. M is the number of objects that were previously classified using criteria $A \ldots Y$ and N is the total number of objects including the new objects with criterion Z. Inequality (6) holds for the general case:

$$lmcAandB \ldots YandZ[M_{A \ldots Y}] \times \frac{(N-M)+(M-X)}{N} < lmcAand \ldots andY[M_{A \ldots Y}] \times \frac{N-M}{N} \qquad (6)$$

The new objects are added to one of the Step 2 subclasses based on the criterion Z, as shown in Step 2 of the Doctris framework. Then Step 3 is repeated based on all criteria $A \ldots Z$, so that the subclasses from Step 2 are refined. For Step 1 there exist two options:

a. Step 1 may not be repeated each time new objects are presented, in which case the Step 1 results remain stable throughout the classification process based on the initial criterion A. Thus, the Step 1 results can serve as a constant basis for the future classification process.

b. Step 1 may be repeated based on the criteria $A \ldots Y$, when new objects are presented with a new criterion Z. Thus, the basis of the classification process could change when new objects are presented. However, this is time consuming and inefficient for classification.

After a series of objects with new criteria have been presented to the algorithm, the size of the subclasses formed in Step 2 will increase beyond an acceptable level. In this case, option b described above should be executed, to decrease the size of the Step 2 subclasses and increase the accuracy of the results.

5 Real Yeast Data

We compared BILCOM to AutoClass and Shrinkage-based hierarchical clustering on yeast data sets of genes with mixed categorical and numerical attribute values (CAs and NAs). We used numerical data derived from gene expression studies on the yeast *Saccharomyces cerevisiae*. These data sets were produced at Stanford to study the yeast cell cycle across time and under various experimental conditions and are available from the SGD database [9, 23]. When clustering this data set, we consider each gene to be an object.

We represented CAs on a gene in terms of Gene Ontology (GO) which is a dynamically controlled vocabulary that can be applied to many organisms, even as knowledge of gene/protein roles in cells changes. GO annotations represent knowledge on genes and are organized along the categories of molecular function, biological process and cellular location [7, 12, 23]. GOSlim are GO annotations that represent higher level knowledge on genes. Most of the GO and GOSlim annotations on the yeast genes exist in the publicly accessible SGD database [8, 12, 20]. We created six pools of CAs for each gene and each pool contained GO annotations of a specific type. Three pools contained GO annotations for molecular function, biological process and cellular location of a gene. The other three pools contained GOSlim annotations for each GO annotation.

5.1 Experiments on Yeast

We have validated BILCOM on the yeast data sets by Cherepinsky et al. [6] shown in Table 1, with mixed categorical and numerical attribute values [7, 9]. We represented CAs on a gene in terms of Gene Ontology (GO) as described above.

However, we perturbed the CAs randomly. This simulates the uncertainty that exists on current knowledge and that is expressed in SGD as GO evidence codes [8, 12, 20]. For this purpose, for each CA we generated a *limit* in a range from 0.1 to 1.0 and, then, generated a random number ρ from 0.0 to 1.0. If ρ exceeded the *limit*, then we perturbed the CA by assigning it a value taken randomly from the set of possible values for that CA.

The yeast microorganism performs a constant cell-cycle. The yeast cell-cycle gene expression program is regulated by the nine known cell-cycle transcriptional activators that control the flow from one stage of the cell-cycle to the next [6]. This regulation of transcriptional activators together with various functional properties suggests a way of partitioning cell-cycle genes into clusters, each one characterized by a group of transcriptional activators working together and their functions [6].

Tables 1 and 2 show two hypotheses about how the genes should be correctly grouped. Table 1 shows grouping by cell-cycle functions. Table 2 shows grouping by stages of the yeast cell-cycle. For instance, by the first hypothesis group 2 is characterized by the activators Swi6 and Mbp1 and the function involving DNA replication and repair at the juncture of G1 and S stages. By the second hypothesis group 2 is characterized by the genes involved in the S stage. The first hypothesis is the same as the one used by Cherepinsky et al. [6], grouping together genes that have the same functions during the cell cycle and are regulated by the same transcriptional activators. The second hypothesis groups together genes that play a prominent role during the same cell-cycle stage.

Table 1. Genes in the data set of Cherepinsky et al. grouped by functions. This is the first hypothesis about the 'correct' grouping of genes.

Group	Activa-tors	Genes	Functions
1	Swi4, Swi6	Cln1, Cln2, Gic1, Msb2, Rsr1, Bud9, Mnn1, Och1, Exg1, Kre6, Cwp1	Budding
2	Swi6, Mbp1	Clb5, Clb6, Rnr1, Rad27, Cdc21, Dun1, Rad51, Cdc45, Mcm2	DNA replication and repair
3	Swi4, Swi6	Htb1, Htb2, Hta1, Hta2, Hta3, Hho1	Chromatin
4	Fkh1	Hhf1, Hht1, Tel2, Arp7	Chromatin
5	Fkh1	Tem1	Mitosis control
6	Ndd1, Fkh2, Mcm1	Clb2, Ace2, Swi5, Cdc20	Mitosis control
7	Ace2, Swi5	Cts1, Egt2	Cytokinesis
8	Mcm1	Mcm3, Mcm6, Cdc6, Cdc46	Prereplication complex formation
9	Mcm1	Ste2, Far1	Mating

Table 2. Genes in our data set grouped by cell-cycle stage. This is the second hypothesis about the 'correct' grouping of genes.

Group	Cell Cycle Stage	Genes	Functions
1	G1	Cln1, Cln2, Gic1, Msb2, Rsr1, Bud9, Mnn1, Och1, Exg1, Kre6, Cwp1	Budding
2	S	Clb5, Clb6, Rnr1, Rad27, Cdc21, Dun1, Rad51, Cdc45, Mcm2	DNA replication and repair
		Htb1, Htb2, Hta1, Hta2, Hta3, Hho1	Chromatin
		Hhf1, Hht1, Tel2, Arp7	Chromatin
3	G2	Tem1	Mitosis control
		Clb2, Ace2, Swi5, Cdc20	Mitosis control
4	M	Cts1, Egt2	Cytokinesis
		Mcm3, Mcm6, Cdc6, Cdc46	Prereplication complex formation
		Ste2, Far1	Mating

Cherepinsky et al. [6] defined a notation to represent the resulting cluster sets and an error scoring function to aid in their comparison. Each cluster set is written as:

$$\{x \rightarrow \{\{y_1, z_1\}, \{y_2, z_2\}, \ldots, \{y_{n_x}, z_{n_x}\}\}\}_{x=1}^{\#\text{ of groups}},$$

where x denotes the group number as described in Table 1, n_x is the number of clusters the members of group x appear in, and for each cluster $j \in \{1,..., n_x\}$ there are y_j genes from group x and z_j genes from other groups in Table 1. A value of * for z_j denotes that cluster j contains additional genes, although none of them are cell-cycle genes. The cluster set can then be scored as follows:

$$\mathrm{FP}(\gamma) = \frac{1}{2}\sum_{x}\sum_{j=1}^{n_x} y_j \cdot z_j$$

$$\mathrm{FN}(\gamma) = \sum_{x}\sum_{1 \le j < k \le n_x} y_j \cdot y_k$$

$$\text{Error score}(\gamma) = \mathrm{FP}(\gamma) + \mathrm{FN}(\gamma).$$

We have compared the error scores of BILCOM on the "perturbed" mixed yeast data set to those of AutoClass [24] on the "perturbed" mixed yeast data set and the Shrinkage-based hierarchical clustering method on the numerical yeast gene

expression data set (a latest algorithm proposed by Cherepinsky et al [6]). As discussed in [6], the Shrinkage-based hierarchical clustering error score for the first hypothesis is 164 and for the second hypothesis it is 264.

Table 3 shows the results for applying AutoClass [24] to the "perturbed" categorical and numerical yeast data set.

Table 3. Clustering results of AutoClass.

Cluster	Genes
1	CLN1, CLN2, GIC1, GIC2, MSB2, RSR1, BUD9, MNN1, OCH1, EXG1, KRE6, CWP1, CLB5, CLB6, RAD51, CDC45, HTB1, HTA2, HHO1, TEL2
2	ARP7, TEM1, CLB2, ACE2, SWI5, CDC20, CTS1, EGT2, MCM3, MCM6, CDC6, CDC46, STE2
3	RNR1, RAD27, CDC21, DUN1, MCM2, HTB2, HTA1, HHF1, HHT1, FAR1

Given the first hypothesis shown in Table 1 and the set of AutoClass results shown in Table 3, the resulting clusters with the error score are written as follows [6]:

{1->{{11,9}}, *2->{{4,16},{5,5}},* *3->{{3,17},{2,8}},* *4->{{1,19},{1,12},{2,8}},* *5->{{1,12}},* *6->{{4,9}},* *7->{{2,11}},* *8->{{4,9}},* *9->{{1,12},{1,9}} }.*	*FP = 265* *FN = 32* *Error = 297*

Given the second hypothesis shown in Table 2 and the set of AutoClass results shown in Table 3, the resulting clusters with the error score are written as follows [6]:

{1->{{11,9}}, *2->{{8,12},{1,12},{9,1}},* *3->{{5,8}},* *4->{{7,6},{1,9}} }.*	*FP = 153* *FN = 96* *Error =249*

Table 4 shows the results for applying BILCOM to the "perturbed" categorical and numerical yeast data set. We produced several sets of results. Because of space limitations we only discuss one set of results here, using as numerical similarity metric the Pearson Correlation coefficient and for a *threshold* value of 1. More experiments for other numerical similarity metrics and different *threshold* values are described in the Appendix.

Table 4. Clustering results of BILCOM using as numerical similarity metric between objects the Pearson correlation coefficient and for a *threshold* value of 1. 25 objects were clustered at the first level.

Cluster	Genes
1	CTS1,EGT2
2	ACE2,SWI5,CDC20,CLB2,TEM1
3	HHO1,ARP7,HHT1
4	RAD27,CDC21,RNR1,OCH1,MNN1,CLN2,DUN1
5	EXG1,CWP1
6	RSR1,BUD9
7	GIC1,TEL2,KRE6,GIC2,MSB2
8	HTB1,HTB2,HTA1,HTA2,HHF1
9	CDC45,MCM2,MCM3,FAR1,CDC6,MCM6,CDC46,STE2
10	CLB5,CLB6,RAD51,CLN1

Given the first hypothesis shown in Table 1 and the set of BILCOM results shown in Table 4, the resulting clusters with the error score are written as follows [6]:

{1->{{3,4},{2,0},{2,0},{4,1},{1,3}}, *2->{{4,3},{3,1},{2,6}},* *3->{{1,2},{4,1}},* *4->{{2,1},{1,4},{1,4}},* *5->{{1,4}},* *6->{{4,1}},* *7->{{2,0}},* *8->{{4,4}},* *9->{{2,6}} }.*	*FP=49* *FN=5+4+26+55=90* *Error = 139*

The BILCOM error of 139 is lower than the Shrinkage-based hierarchical clustering error of 164 and the AutoClass error of 297 for the first hypothesis.

Given the second hypothesis shown in Table 2 and the set of BILCOM results shown in Table 4, the resulting clusters with the error score are written as follows [6]:

{1->{{3,4},{2,0},{2,0},{4,1},{1,3}}, *2->{{3,0},{4,3},{1,4},{5,0},{3,1},{2,6}},* *3->{{5,0}},* *4->{{2,0},{6,2}} }.*	*FP=31* *FN=12+55+130* *Error=228*

The BILCOM error of 228 is lower than the Shrinkage-based hierarchical clustering error of 264 and the AutoClass error of 249 for the second hypothesis.

The error scores are summarized in Table 5 below. The BILCOM error rate is lower than AutoClass and Shrinkage-based hierarchical clustering, for both hypotheses in Tables 1 and 2. BILCOM clusters are closer to the desired groupings.

Table 5. Comparative error rates of algorithms applied to the yeast data set.

Clustering Algorithm	First hypothesis	Second hypothesis
BILCOM	139	228
Shrinkage-based hierarchical	164	264
AutoClass	297	249

6 Hepatitis and Thyroid Disease Data

Given a biomedical disease data set, predicting which patients will live or die may be tackled as a supervised learning problem involving finding a hyperspace separator between patients of the 'DIE' and 'LIVE' class, based on a set of training cases with known outcomes. However, such a hyperspace separator is often not trivial, as Figure 8 shows. C*lustering* methods can be used instead to cluster the patients into 'DIE' and 'LIVE' groups. We apply BILCOM and other clustering algorithms to cluster the *hepatitis* and *thyroid disease* data sets of mixed categorical and numerical attribute values (CAs and NAs) from the UCI Irvine Machine Learning Repository [24]. Tables 6 and 7 describe these data sets. The objects in both data sets are patients with class labels that enable us to compare our clustering results with the true classes. Class labels were removed from the objects before clustering and no information about the true classes was given to the process.

The hepatitis data set has 155 objects with 13 CAs and 6 NAs. The objects are split into 2 classes: 'DIE' and 'LIVE'. Of the 155 objects, 32 belong to the 'DIE' class and 123 to the 'LIVE' class.

The thyroid disease data set has 3163 objects with 12 CAs and 7 NAs. The objects are split into 2 classes: 'hypothyroid' and 'negative'. Of the 3163 objects, 151 belong to the 'hypothyroid' class and 3012 to the 'negative' class. For the *thyroid disease* data set we make the clustering challenge harder by perturbing about half of all CAs before clustering, turning 'true' to 'false' and 'false' to 'true'. The reason we perturbed the CAs was to show that if little categorical similarity can be found between an object and a cluster at the first level, then the object will be clustered at the second level based on numerical similarity, increasing its chance to be clustered correctly.

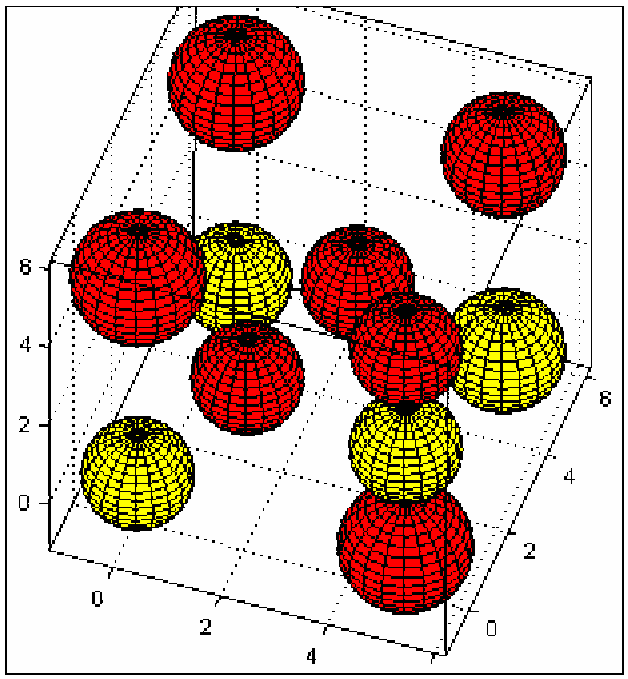

Fig. 8. Clusters in a 3-dimensional biomedical disease data set containing objects (patients) who will live or die. The red clusters contain patients of the 'LIVE' class. The yellow clusters contain patients of the 'DIE' class. As shown, this data set is unbalanced since the 'LIVE' patients outnumber the 'DIE' patients. It would be hard to find a hyperspace separator between 'LIVE' and 'DIE' patients.

Table 6. The *Hepatitis* data set (155 objects). Classes: DIE (32 objects), LIVE (123 objects).	
13 CAs - represent something observed to be true about a hepatitis patient.	6 NAs –represent the results of an experiment on a hepatitis patient.
SEX: male, female	Bilirubin: real
Steroid: no, yes	Alk Phosphate: real
Antivirals: no, yes	SGOT: real
Fatigue: no, yes	Albumin: real
Malaise: no, yes	Protime: real
Anorexia: no, yes	Age: integer
Liver big: no, yes	
Liver firm: no, yes	
Spleen Palpable: no, yes	
Spiders: no, yes	
Ascites: no, yes	
Varices: no, yes	
Histology: no, yes	

Table 7. The *Thyroid disease* data set (3163 objects). Classes: hypothyroid (151 objects), negative (3012 objects).	
12 CAs - represent something observed to be true about a thyroid patient.	7 NAs –represent the results of an experiment on a thyroid patient.
SEX: male, female	TSH: real
On thyroxine: no, yes	T3: real
Query on thyroxine: no, yes	TT4: real
On antithyroid medication: no, yes	T4U: real
Thyroid surgery: no, yes	FTI: real
Query hypothyroid: no, yes	TBG: real
Query hyperthyroid: no, yes	Age: integer
Pregnant: no, yes	
Sick: no, yes	
Tumor: no, yes	
Lithium: no, yes	
Goitre: no, yes	

Our misclassification rate measure is the *classes to clusters evaluation* that is used by the clustering algorithms of the WEKA package [25,28]. In this mode we first ignore the class attribute and generate the clusters. Then during the test phase we assign classes to the clusters, based on the majority value of the class attribute within each cluster. Then we compute the classification error, based on this assignment.

6.1 Comparison of BILCOM to other Algorithms for the Hepatitis and Thyroid Disease data sets

We cluster hepatitis and thyroid disease data sets of mixed CAs and NAs with BILCOM, AutoClass [27], as well as k-Means [21] treating the CAs as NAs. Then we split each of the data sets into numerical and categorical data types and we cluster each type separately. We cluster the numerical type with k-Means. We cluster the categorical type with k-Modes [21] and MULIC [2]. For k-Modes and k-Means we set the number of clusters to the number of 'true' classes in the data sets of 2 and the convergence threshold to 0. We also experimented with a number of clusters larger than 2 but the misclassification rate did not change significantly. The modes of the initial clusters are set equal to the values of the first objects inserted in the clusters. For AutoClass, k-Modes, k-Means and MULIC we run 5 random starts on each data set with different orderings of objects and we report the average result, since these algorithms may produce different results for different orderings. For AutoClass we did not specify the number of clusters as the software considers results for numbers of clusters varying from 2 to 35; we set the prior distribution for the attributes to the *single multinomial distribution*, with no attributes ignored, which was also the distribution chosen by the developers of the software for their tests on the soybean data sets [24].

Tables 8 and 9 compare the accuracy of the results for all algorithms. The hepatitis disease data set contains 2 classes - 'DIE' and 'LIVE' - and the first class has 32 objects while the second class has 123 objects, implying that the misclassification rate is likely to be between 0 and 32/155. The BILCOM misclassification rate is lower than this at 17/155. When clustering the hepatitis disease data set with BILCOM, the average ratio of 'DIE' to 'LIVE' objects across all clusters in which at least one 'DIE' object appears is 32/64 = 50%, which is higher than the ratio of 'DIE' to 'LIVE' objects for the entire data set of 32/123 = 26%. When clustering with k-Modes or AutoClass, this average ratio is 32/95 = 33%. When clustering with MULIC inputting just the CAs of each object, this average ratio is 32/76 = 42%. This supports that BILCOM separates the objects of the minority 'DIE' class in such an imbalanced data set, better than other algorithms. This suggests that if an unannotated patient z is clustered together with at least one other 'DIE' patient then patient z is 50% likely to die.

Table 8. Clustering algorithms and misclassification rates for the *hepatitis* data set.	
AutoClass for 2 clusters taking as input CAs and NAs	32/155 = 20.64%
k-Modes for 2 clusters taking as input CAs	32/155 = 20.64%
k-Means for 2 clusters taking as input NAs	32/155 = 20.64%
k-Means for 2 clusters taking as input CAs and NAs	25/155 = 16%
MULIC taking as input CAs	20/155 = 12.9%
BILCOM taking as input CAs and NAs	17/155 = 10.9%

Table 9. Clustering algorithms and misclassification rates for *thyroid disease* data set.	
AutoClass for 2 clusters taking as input CAs and NAs	151/3163 = 4.77%
k-Modes for 2 clusters taking as input CAs	151/3163 = 4.77%
k-Means for 2 clusters taking as input NAs	151/3163 = 4.77%
k-Means for 2 clusters taking as input CAs and NAs	145/3163 = 4.58%
MULIC taking as input CAs	138/3163 = 4.36%
BILCOM taking as input CAs and NAs	130/3163 = 4.11%

The thyroid disease data set contains 2 classes - 'hypothyroid' and 'negative' - and the first class has 151 objects while the second class has 3012 objects, implying that the misclassification rate is likely to be between 0 and 151/3163. The BILCOM misclassification rate is lower than this at 130/3163. When clustering the thyroid disease data set with BILCOM, the average ratio of 'hypothyroid' to 'negative' objects across all clusters in which at least one 'hypothyroid' object appears is 151/755=20%, which is higher than the ratio of 'hypothyroid' to 'negative' objects for the entire data set of 151/3012 = 5%. When clustering with k-Modes or AutoClass, this average ratio is 151/2200 = 6.8%. When clustering with MULIC inputting just the CAs of each object, this average ratio is 151/1520 = 9.9%. This supports that BILCOM separates the objects of the minority 'hypothyroid' class in such an imbalanced data set, better than other algorithms. This suggests that if an unannotated patient *z* is clustered together with at least one other 'hypothyroid' patient then patient *z* is 20% likely to be thyroid positive.

Table 10. The average percentage of misclassified objects increases in bottom layers of MULIC clusters.			
Hepatitis	Misclassifications	Thyroid disease	Misclassifications
Layer 0	2%	Layer 0	3%
Layer 1	5%	Layer 1	20%
Layer 2	45%	Layer 2	30%
Layer 3	50%		

Table 11. MULIC average misclassification rates for the objects clustered in layers of depth greater than the *threshold* value (0 for hepatitis and 0 for thyroid data sets).	
Hepatitis data set	15/75 = 20%
Thyroid disease data set	120/1700 = 9.2%

We cluster the hepatitis and thyroid disease data sets with MULIC [2] inputting just the CAs of each object. Table 10 shows that in bottom layers of MULIC clusters, the average percentage of objects misclassified, i.e., placed in a wrong cluster, increases. Table 11 shows the average misclassification rates for MULIC in layers of depth greater than the *threshold* value, which is 0 for hepatitis and 0 for thyroid disease. As we find out, the MULIC misclassification rate is higher in these layers than the BILCOM misclassification rate. This supports clustering at the first level using categorical similarity the objects in layers of depth less than or equal to the *threshold* value, while clustering the other objects at the second level using numerical similarity.

6.2 Discussion of Results for the Hepatitis Data Set

In our experiments with the hepatitis data set, the number of objects participating at the first level is 76, while the number of objects participating at the second level is 79. At the second level there are 53 center-containing subclusters and 23 seed-containing subclusters, implying a total of 23 clusters.

Despite the imbalanced hepatitis data set classes, most BILCOM clusters produced have either a strong majority of 'DIE' objects or a strong majority of 'LIVE' objects. For example, the 3 clusters shown in Table 12 contain a strong majority of 'DIE' objects, showing that the algorithm is able to separate 'DIE' objects from 'LIVE' objects, even though 'DIE' objects compose a minority ratio of 32/155 of the objects in the hepatitis data set.

Table 12. Three clusters resulting from clustering the *hepatitis* data set with BILCOM, containing a majority of 'DIE' objects. The numerical similarity metric between objects is the average distance over all pairs of numerical attributes and the *threshold* value is 0.
Cluster 1: Subcluster 1.1: DIE Subcluster 1.2: DIE
Cluster 2: Subcluster 2.1: DIE, DIE, DIE, DIE, DIE, DIE, DIE, DIE, DIE, DIE, LIVE, LIVE, LIVE Subcluster 2.2: LIVE
Cluster 3:

Subcluster 3.1: DIE, DIE, DIE, DIE, DIE, DIE, DIE, LIVE Subcluster 3.2: DIE, LIVE

Many clusters produced by BILCOM contain only or mostly 'LIVE' objects. For example, the largest cluster produced contains 18 subclusters and each subcluster contains between 1 and 3 objects *all* of which belong to the 'LIVE' class.

There are several cases where objects are assigned to a different cluster at the second level from what the first level results suggested. This might have been caused because an object had a low categorical similarity to the mode of its first level cluster, or because it was assigned erroneously to its first level cluster and its numerical similarity to its second level cluster is stronger. Table 13 shows that in our experiments with the hepatitis data set, 18 center-containing subclusters (out of 53) containing 38 objects in total, end up being merged to a different cluster from what the first level results suggested. The other 35 center-containing subclusters are merged to the same cluster as the first level results suggested. Four of these 18 center-containing subclusters have a majority of 'DIE' objects. The percentage of objects in these 18 center-containing subclusters that are attached to the wrong cluster is 10%, based on whether 'LIVE' or 'DIE' is most prominent in the cluster.

When comparing Column 7 to Tables 10 and 11, the BILCOM misclassification rate (derived by subtracting Column 7 from 100) is slightly lower than the MULIC misclassification rate for the objects clustered at layers greater than the value of *threshold*; BILCOM clustered these objects based on numerical rather than categorical similarity.

For many center-containing subclusters, as the looping of step 5 of the second level progresses, their numerical similarity to their most numerically similar seed-containing subclusters determined at step 4 decreases – this is a sign that their tendency to get merged to the clusters suggested by the first level becomes stronger. For example, in an earlier loop of step 5 the numerical similarities of several center-containing subclusters to their most numerically similar seed-containing subclusters have the values 0.121951, but in the last loop of step 5 they have weaker similarities of 0.11875.

Table 13. 18 center-containing subclusters (out of 53) containing 38 objects in total, end up being merged to a different cluster from what the first level cluster suggests. For all of these subclusters, column3×column4 < column5×column6.

Column 1: Center-containing subcluster	Column 2: Most prominent class in the center-containing subcluster	Column 3: Categorical similarity of center to seed of its 1st level cluster in the last step 5 loop	Column 4: Numerical similarity of center-containing subcluster to its seed-containing cluster from 1st level in the last step 5 loop	Column 5: Categorical similarity of center to seed of its 2nd level seed-containing most numerically similar cluster in the last step 5 loop	Column 6: Numerical similarity of center-containing subcluster to its 2nd level seed-containing most numerically similar cluster in the last step 5 loop	Column 7: Percentage of objects in center-containing subcluster that were attached to the correct cluster
1	LIVE	0.785714	0.0833333	0.857143	0.11875	75%
2	DIE	0.857143	0.0625	0.642857	0.11875	90%
3	LIVE	0.857143	0.0625	0.642857	0.11875	80%
4	DIE	0.928571	0.0833333	0.857143	0.11875	90%
5	LIVE	0.928571	0.333333	0.857143	0.475	100%
6	DIE	0.928571	0.333333	0.857143	0.475	90%
7	LIVE	0.928571	0.111111	0.857143	0.158333	85%
8	LIVE	0.5	0.333333	0.571429	0.475	90%
9	LIVE	0.928571	0.333333	0.714286	0.475	75%
10	LIVE	0.928571	0.111111	0.714286	0.158333	90%
11	LIVE	0.928571	0.111111	0.714286	0.158333	100%
12	LIVE	0.857143	0.333333	0.785714	0.475	80%
13	LIVE	0.785714	0.2	0.714286	0.475	80%
14	LIVE	0.785714	0.2	0.714286	0.475	90%
15	DIE	0.785714	0.0666667	0.714286	0.158333	75%
16	LIVE	0.785714	0.2	0.714286	0.475	100%
17	LIVE	0.785714	0.2	0.714286	0.475	90%
18	LIVE	0.928571	0.2375	0.928571	0.2375	80%

6.3 Runtime Evaluation of BILCOM

Tables 14-15 compare BILCOM's runtime to AutoClass [27], k-Modes [21] and MULIC [2] on two data sets. All of these algorithms are implemented in C/C++. The experiments were performed on a Sun Ultra 60 with 256 MB of memory and a 300 MHz processor. BILCOM often executes faster by decreasing the value of *minimum_numerical_similarity* at the second level, or decreasing *threshold* or $\delta\varphi$ at the first level [1,2]. Section 7 discusses selecting values for these parameters.

Table 14. Seconds for clustering the *hepatitis* data set	
AutoClass	0.24
k-Modes	0.01
MULIC	0
BILCOM	0.02

Table 15. Seconds for clustering the *thyroid disease* data set	
AutoClass	8.24
k-Modes	1.13
MULIC	0.49
BILCOM	1.14

7 Selecting BILCOM Parameters

The objects clustered at the first level are those whose similarity to the closest mode is greater than or equal to the value of *minimum_mode_similarity*, translating to a *threshold* value of *m-minimum_mode_similarity*. One could choose a quarter of all objects to participate at the first level so that each second level subcluster would have on average 4 objects. We do not use this approach because it is more reasonable for the user to select the first level objects based on their similarity to modes, since the objects with high similarity are more likely to be classified in the correct cluster. For example, Tables 10-11 showed that when clustering the hepatitis and thyroid disease data sets with MULIC inputting categorical data only, at lower layers of a cluster the percentage of objects misclassified increases.

We experiment with various values of *minimum_mode_similarity* for separating first and second level objects, for the hepatitis data set. The table below shows the resulting changes in the number of objects that are clustered at the first level, as well as the average size of the second level subclusters. The lower the value of *minimum_mode_similarity*, the more objects are clustered at the first level and the lower the average size of the second level subclusters. Thus, the value should be neither too high nor too low. For hepatitis we use a *minimum_mode_similarity* value of 12, translating to a *threshold* value of 1.

minimum_mode_similarity	1	10	15	18
Number of objects clustered at first level	155	142	110	76
Average size of second level subclusters	1	1.09	1.4	2.12

The values of the variables in the loop of step 5 of the second level depend on the value of *minimum_numerical_similarity*, used for creating links between objects at step 4. If the value of *minimum_numerical_similarity* is too low, then there will be links created between most of the objects and many subclusters will be numerically similar to one another. On the other hand, if the value is high then there will be fewer links created and fewer subclusters will be numerically similar to one another.

We experiment with various values of *minimum_numerical_similarity* for the hepatitis data set, to determine how many center-containing subclusters remain merged to their numerically closest seed-containing subcluster after step 5 of the second level. The table below shows the results. With a low value more center-containing subclusters remain merged to the numerically closest seed-containing subcluster to which they were merged in step 4 of the second level. This value can be chosen by the user, depending on how s/he wants to distribute the classification of objects based on numerical similarity or categorical similarity. For hepatitis we use a *minimum_numerical_similarity* value of 0.5.

minimum_numerical_similarity	0.1	0.3	0.5	0.7	0.9
Number of subclusters that remain merged to their numerically closest seed-containing subcluster after step 5 of the second level	40	39	35	33	31

Figure 9 is a graph showing what values the variables in the inequality comparison of step 5 would need to take, for a center-containing subcluster to remain merged to its numerically similar seed-containing second level subcluster, instead of the subcluster suggested by level one. This graph assumes a value of 1.0 for the numerical similarity of the center-containing subcluster to its most numerically similar second level seed-containing subcluster - for lower values of this variable the shape of the figure looks the same, except that the y axis has a higher range.

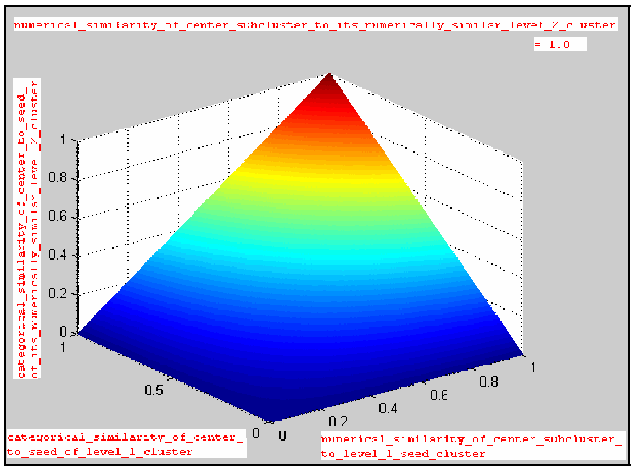

Fig. 9. This graph illustrates the values that the variables in the inequality comparison of second level step 5 would need to take, for a center-containing subcluster to remain merged to its numerically similar seed-containing second level subcluster, instead of the subcluster suggested by level one. This graph assumes a value of 1.0 for the numerical similarity of the center-containing subcluster to its most numerically similar second level seed-containing subcluster. The x-z axis shows the categorical and numerical similarity of the center-containing subcluster to the first level cluster in which the center was clustered. The y axis shows how high the categorical similarity of the center to the seed in the most numerically similar seed-containing second level subcluster would need to be for the center-containing subcluster to remain merged to it.

8 Conclusion

In analyzing biological data, it is important to include all of the existing information into the analysis process. The BILCOM clustering algorithm gives the 'full picture' of a data set by using a mix of two data types: categorical and numerical data types. This algorithm is inspired by Bayesian classification theory [4] and uses categorical clustering as a prior to maximize the probabilities that objects will be assigned to the correct clusters. We have tested BILCOM's accuracy against other algorithms that cluster mixed and non-mixed types. BILCOM's runtime is comparable to other algorithms.

Physicians will find this technique useful in the field of evidence-based medicine, for deriving conclusions about the outcome of a patient's condition based on evidence from other patients' conditions' outcomes. In our example of clustering hepatitis patient data, there were clusters that contained a majority of objects of class 'DIE' even though this class occurred infrequently in the data set overall. If a new unknown object gets clustered in a cluster with many other objects of class 'DIE', a physician could derive conclusions about a patient's condition's future outcome.

Biologists will also find this method useful in wet lab work, for deriving hints about the potential functions of genes and proteins. In [3] we discussed significance metrics to identify the most significant functional annotations in a cluster and apply them to other genes classified in the same cluster, for which less or no functional knowledge exists. Many genes have little or no knowledge associated with them. The hints that are derived about a gene's function can be validated experimentally.

References

[1] Andreopoulos, B., An, A. and Wang, X. (2005) BILCOM: Bi-level Clustering of Mixed Categorical and Numerical Biological Data. Technical report CS-2005-01. Department of Computer Science and Engineering, York University, January 2005.

[2] Andreopoulos, B., An, A. and Wang, X. (2004) MULIC: Multi-Layer Increasing Coherence Clustering of Categorical Data Sets. Technical report CS-2004-07. Department of Computer Science and Engineering, York University, December 2004.

[3] Andreopoulos, B., An, A. and Wang, X. Clustering Mixed Numerical and Uncertain Categorical Data with BILCOM: Significance Metrics on a Yeast Example. Technical report CS-2005-03. York University, Department of Computer Science and Engineering. March 2005.

[4] Andreopoulos, B., An, A. and Wang, X. (2005) A Framework for Unsupervised Learning with Multiple Criteria. Technical report CS-2005-10. Department of Computer Science and Engineering, York University.

[5] P. Andritsos, P. Tsaparas, R. J. Miller, K. C. Sevcik. LIMBO: Scalable Clustering of Categorical Data. In 9th International Conference on Extending DataBase Technology (EDBT) 2004.

[6] Cherepinsky V., Feng J., Rejali M., and Mishra B. (2003) Shrinkage-based similarity metric for cluster analysis of microarray data. PNAS, Vol. 100, Issue 17, 9668-9673.

[7] Dwight SS, Harris MA, Dolinski K, Ball CA, Binkley G, Christie KR, Fisk DG, Issel-Tarver L, Schroeder M, Sherlock G, Sethuraman A, Weng S, Botstein D, Cherry JM. Saccharomyces Genome Database provides secondary gene annotation using the Gene Ontology. *Nucleic Acids Research* 30: 69-72. 1999.

[8] Eisen, M.B. & Brown, P.O. (1999) DNA arrays for analysis of gene expression. *Methods Enzymol.* 303, 179-205.

[9] Eisen MB, Spellman PT, Brown PO, Botstein D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci USA. 1998 Dec 8;*95(25):14863-8.
[10] Fasulo D. (1999) An Analysis of Recent Work on Clustering Algorithms, Technical Report # 01-03-02, Department of Computer Science & Engineering, University of Washington.
[11] V. Ganti, J. Gehrke, R. Ramakrishnan. CACTUS-clustering categorical data using summaries. In Proc of KDD '99, pp. 73-83.
[12] The Gene Ontology Consortium. (2001). Creating the gene ontology resource: design and implementation. *Genome Research* 11: 1425-1433.
[13] D. Gibson, J. Kleiberg, P. Raghavan. Clustering categorical data: an approach based on dynamic systems. In Proc of VLDB 1998, pp. 311-323.
[14] Goebel, M. & Gruenwald, Le (1999). A survey of data mining and knowledge discovery software tools. ACM SIGKDD Explorations 1, 20-33.
[15] Golub, T. R. et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* 286, 531-537.
[16] Grambeier J., Rudolph A. (2002) Techniques of Cluster Algorithms in Data Mining. Data Mining and Knowledge Discovery 6: 303-360.
[17] Guha S., Rastogi R., Shim K. (2000). ROCK: A Robust Clustering Algorithm for Categorical Attributes. Information Systems 25(5): 345-366.
[18] Hartigan, J. A. (1975) Clustering algorithms. (John Wiley and Sons, New York, 1975).
[19] Wu L.F., Hughes T.R., Davierwala A.P., Robinson M.D., Stoughton R. and Altschuler S.J. (2002) Large-scale Prediction of Saccharomyces Cerevisiae Gene Function Using Overlapping Transcriptional Clusters. *Nature Genetics* 31:255-265.
[20] Huang Z., Ng M.K.. (1999) A fuzzy k-modes algorithm for clustering categorical data. IEEE Transaction on Fuzzy Systems, 7(4): 446-452.
[21] Huang Z. (1998) Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Knowledge Discovery 2(3): 283-304.
[22] Huang, Z. (1997) Clustering Large Data Sets with Mixed Numeric and Categorical Values. Knowledge discovery and data mining: techniques and applications. World Scientific.
[23] Lord P.W., Stevens R.D., Brass A. and Goble C.A. (2003). Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics* 19: 1275-83.
[24] C.J.Mertz, P.Merphy. "UCI Repository of Machine Learning Databases", 1998, http://www.ics.uci.edu/~mlearn.
[25] P. Reutemann, B. Pfahringer, E. Frank. (2004) Proper: A Toolbox for Learning from Relational Data with Propositional and Multi-Instance Learners. 17th Australian Joint Conference on Artificial Intelligence (AI2004). Springer-Verlag
[26] Slonim D.K., Tamayo P., Mesirov J.P., Golub T.R., and Lander E.S.. (2000) Class prediction and discovery using gene expression data. *Proceedings of the Fourth Annual Conference on Computational Molecular Biology (RECOMB),* 263-272.
[27] Stutz J. and Cheeseman P. (1995) Bayesian Classification (AutoClass): Theory and results. Advances in Knowledge Discovery and Data Mining, 153-180, AAAI Press.
[28] Witten I. and Frank E. "Data Mining: Practical machine learning tools with Java implementations". Morgan Kaufmann, San Francisco, 2000.
[29] Christopher J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery. Volume 2, Issue 2. Pages: 121–167. (June 1998).
[30] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

Appendix – Detailed BILCOM Results on Yeast

We produced 4 sets of results for BILCOM. Table *a* shows our results for using as numerical similarity metric between two objects the average distance over all pairs of numerical attributes and for a *threshold* value (maximum value for φ) of 11. Table *b* shows our results for using as numerical similarity metric between two objects the Pearson correlation coefficient and for a *threshold* value of 11. Table *c* shows our results for using as numerical similarity metric between two objects the Pearson correlation coefficient and for a *threshold* value of 1. Table *d* shows our results for using as numerical similarity metric between two objects the average distance over all pairs of numerical attributes and for a *threshold* value of 1.

Table a. Clustering results of BILCOM using as numerical similarity metric between 2 objects the average distance over all pairs of numerical attributes and for a *threshold* value of 11. 35 objects were clustered at the first level.

Cluster	Genes
1	CLB2,CLN2,CDC21,FAR1
2	CTS1,EGT2
3	ACE2,CDC20,SWI5
4	ARP7,TEM1,HHO1,HHT1
5	RAD27,DUN1

6	KRE6,TEL2,EXG1,CWP1
7	RSR1,BUD9
8	GIC1,GIC2,MSB2
9	HTB1,HTB2,HTA1,HTA2,HHF1
10	RNR1,MNN1,CDC6,CDC45,MCM2,STE2,MCM3,MCM6,CDC46
11	CLB5,RAD51,OCH1,CLB6,CLN1

Table b. Clustering results of BILCOM using as numerical similarity metric between 2 objects the Pearson correlation coefficient and for a *threshold* value of 11. 35 objects were clustered at the first level.

Cluster	Genes
1	CDC21,CLN2,RAD51,CLB2,CDC20,FAR1,STE2
2	CTS1,EGT2
3	ACE2,SWI5,TEM1
4	ARP7,HHO1,HHT1
5	RAD27,OCH1,MNN1,DUN1
6	KRE6,EXG1,CWP1
7	RSR1,BUD9
8	GIC1,TEL2,GIC2,MSB2
9	HTB1,HTB2,HTA1,HTA2,HHF1
10	RNR1,CDC6,CDC45,MCM2,MCM3,MCM6,CDC46
11	CLB5,CLB6,CLN1

Table c. Clustering results of BILCOM using as numerical similarity metric between 2 objects the Pearson correlation coefficient and for a *threshold* value of 1. 25 objects were clustered at the first level.

Cluster	Genes
1	CTS1,EGT2
2	ACE2,SWI5,CDC20,CLB2,TEM1
3	HHO1,ARP7,HHT1
4	RAD27,CDC21,RNR1,OCH1,MNN1,CLN2,DUN1
5	EXG1,CWP1
6	RSR1,BUD9
7	GIC1,TEL2,KRE6,GIC2,MSB2
8	HTB1,HTB2,HTA1,HTA2,HHF1
9	CDC45,MCM2,MCM3,FAR1,CDC6,MCM6,CDC46,STE2
10	CLB5,CLB6,RAD51,CLN1

Table d. Clustering results of BILCOM using as numerical similarity metric between 2 objects the average distance over all pairs of numerical attributes and for a *threshold* value of 1. 25 objects were clustered at the first level.

Cluster	Genes
1	CTS1,EGT2
2	ACE2,CDC20,SWI5,CLB2
3	HHO1,HHT1
4	RAD27,CDC21,RNR1,MNN1,DUN1
5	EXG1,CWP1
6	RSR1,BUD9
7	GIC1,ARP7,TEL2,KRE6,GIC2,MSB2
8	HTB1,HTB2,HTA1,HTA2,HHF1
9	CDC45,MCM2,STE2,MCM3,FAR1,CDC6,MCM6,TEM1,CDC46

10	CLB6,CLB5,RAD51,OCH1,CLN1,CLN2

Cherepinsky et al. [6] defined a notation to represent the resulting cluster sets and an error scoring function to aid in their comparison. Each cluster set is written as:

$$\{x \rightarrow \{\{y_1, z_1\}, \{y_2, z_2\}, \ldots, \{y_{n_x}, z_{n_x}\}\}\}_{x=1}^{\#\text{ of groups}},$$

where x denotes the group number (as described in Tables 1 and 2), n_x is the number of clusters group x appears in, and for each cluster $j \in \{1,..., n_x\}$ there are y_j genes from group x and z_j genes from other groups in Tables 1 and 2. A value of * for z_j denotes that cluster j contains additional genes, although none of them are cell-cycle genes. The cluster set can then be scored according to the following measure:

$$\mathrm{FP}(\gamma) = \frac{1}{2}\sum_{x}\sum_{j=1}^{n_x} y_j \cdot z_j$$

$$\mathrm{FN}(\gamma) = \sum_{x}\sum_{1 \le j < k \le n_x} y_j \cdot y_k$$

$$\text{Error score}(\gamma) = \mathrm{FP}(\gamma) + \mathrm{FN}(\gamma).$$

We have compared the error scores of BILCOM on the "perturbed" mixed yeast data set to those of the Shrinkage-based hierarchical clustering method on the numerical yeast gene expression data set (a latest algorithm proposed by Cherepinsky et al [6]). As discussed in [6], the Shrinkage-based hierarchical clustering error score for the first hypothesis is 164 and for the second hypothesis it is 264.

Given the first hypothesis (Table 1) and the set of BILCOM results shown in Table *c*, the resulting clusters with the corresponding error score are written as follows:

{1->{{3,4},{2,0},{2,0},{4,1},{1,3}}, *2->{{4,3},{3,1},{2,6}},* *3->{{1,2},{4,1}},* *4->{{2,1},{1,4},{1,4}},* *5->{{1,4}},* *6->{{4,1}},* *7->{{2,0}},* *8->{{4,4}},* *9->{{2,6}}* *}.*	*FP=49* *FN=5+4+26+55=90* *Error = 139*

This is better than the Shrinkage-based hierarchical clustering error for the first hypothesis of 164.

Given the second hypothesis (Table 2) and the set of BILCOM results shown in Table *c*, the resulting clusters with the corresponding error score are written as follows:

{1->{{3,4},{2,0},{2,0},{4,1},{1,3}}, *2->{{3,0},{4,3},{1,4},{5,0},{3,1},{2,6}},* *3->{{5,0}},* *4->{{2,0},{6,2}}* *}.*	*FP=31* *FN=12+55+130* *Error=228*

This is better than the Shrinkage-based hierarchical clustering error for the second hypothesis of 264.

Given the first hypothesis (Table 1) and the set of BILCOM results shown in Table *d*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,4},{2,0},{2,0},{4,2},{3,3}}, *2->{{4,1},{2,7},{3,3}},* *3->{{1,1},{4,1}},* *4->{{1,1},{2,4},{1,4}},* *5->{{1,8}},* *6->{{4,0}},* *7->{{2,0}},* *8->{{4,5}},* *9->{{2,7}},* *}.*	*FP = 54* *FN = 5+4+26+55 = 90* *Error=144*

This is better than the Shrinkage-based hierarchical clustering error for the first hypothesis of 164.

Given the second hypothesis (Table 2) and the set of BILCOM results shown in Table *d*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,4},{2,0},{2,0},{4,2},{3,3}}, *2->{{2,0},{4,1},{2,4},{5,0},{3,3},{2,7}},* *3->{{4,0},{1,8}},* *4->{{2,0},{6,3}},* *}.*	*FP = 41* *FN = 12+4+131+55 = 202* *Error = 243*

This is better than the Shrinkage-based hierarchical clustering error for the second hypothesis of 264.

Given the first hypothesis (Table 1) and the set of BILCOM results shown in Table *b*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,6},{2,2},{3,0},{2,0},{3,1},{1,2}}, *2->{{2,5},{2,2},{2,1},{3,4}},* *3->{{1,2},{4,1}},* *4->{{2,1},{1,3},{1,4}},* *5->{{1,2}},* *6->{{2,5},{2,1}},* *7->{{2,0}},* *8->{{4,3}},* *9->{{2,5}}* *}.*	*FP=47* *FN=4+5+4+30+58=101* *Error = 148*

This is better than the Shrinkage-based hierarchical clustering error for the first hypothesis of 164.

Given the second hypothesis (Table 2) and the set of BILCOM results shown in Table *b*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,2},{3,1},{2,0},{3,0},{2,2},{1,6}}, *2->{{2,1}{3,4},{5,0},{1,3},{2,2},{3,0},{2,5}},* *3->{{3,0},{2,5}},* *4->{{4,3},{2,0},{2,5}}* *}.*	*FP=39* *FN=20+6+134+58=218* *Error=257*

This is better than the Shrinkage-based hierarchical clustering error for the second hypothesis of 264.

Given the first hypothesis (Table 1) and the set of BILCOM results shown in Table *a*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,3},{3,1},{2,0},{3,0},{2,3},{1,8}}, *2->{{1,3},{2,0},{3,2},{3,6}},* *3->{{1,3},{4,1}},* *4->{{2,2},{1,3},{1,4}},* *5->{{1,3}},* *6->{{1,3},{3,0}},* *7->{{2,0}},* *8->{{4,5}},* *9->{{1,3},{1,8}}* *}.*	*FP = 50* *FN = 13+29+55 = 97* *Error = 147*

This is better than the Shrinkage-based hierarchical clustering error for the first hypothesis of 164.

Given the second hypothesis (Table 2) and the set of BILCOM results shown in Table *a*, the resulting clusters with the corresponding error score are written as follows:

{1->{{1,3},{3,1},{2,0},{3,0},{1,8},{2,3}}, *2->{{1,3},{3,1},{2,0},{1,3},{5,0},{3,6},{3,2}},* *3->{{1,3},{3,0},{1,3}},* *4->{{1,3},{2,0},{5,4}}* *}.*	*FP = 41* *FN = 58+7+17+133 = 215* *Error = 256*

This is better than the Shrinkage-based hierarchical clustering error for the second hypothesis of 264.

The error scores support that BILCOM was better able to identify the sought after clusters. The best results are shown in Table *c*, where the threshold value is set to 1.