From Context to Distance: Learning Dissimilarity for Categorical Data Clustering

DINO IENCO, RUGGERO G. PENSA and ROSA MEO
Dept. of Computer Science, University of Torino, Italy

Clustering data described by categorical attributes is a challenging task in data mining applications. Unlike numerical attributes, it is difficult to define a distance between pairs of values of a categorical attribute, since the values are not ordered. In this paper, we propose a framework to learn a context-based distance for categorical attributes. The key intuition of this work is that the distance between two values of a categorical attribute A_i can be determined by the way in which the values of the other attributes A_j are distributed in the dataset objects: if they are similarly distributed in the groups of objects in correspondence of the distinct values of A_i a low value of distance is obtained. We propose also a solution to the critical point for the choice of the attributes A_j. We validate our approach by embedding our distance learning framework in a hierarchical clustering algorithm. We applied it on various real world and synthetic datasets, both low and high-dimensional. Experimental results show that our method is competitive w.r.t. the state of the art of categorical data clustering approaches. We also show that our approach is scalable and has a low impact on the overall computational time of a clustering task.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*data mining*; I.5.1 [**Pattern Recognition**]: Clustering—*similarity measures*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: categorical data, clustering, distance learning

1. INTRODUCTION

Clustering is a popular data mining technique that enables to partition data into groups or clusters in such a way that objects inside a group are similar, and objects belonging to different groups are dissimilar [Han and Kamber 2000]. Clearly, the notion of similarity is central in such a process. When objects are described by numerical (real, integer) features, there is a wide range of possible choices. Among them, probably the most popular metric is Euclidean distance (or 2-norm distance), which is a special case of Minkowski distance (also called p-norm distance). Commonly objects can be considered as vectors in a n-dimensional space, where n is the number of features. Given two objects, the distance measure between them only depends on the difference between the values of the feature vectors.

In data mining applications, however, data are often described by categorical attributes

Authors' address: Università di Torino, Dipartimento di Informatica, Corso Svizzera 185, I-10149 Torino, Italy, e-mail: ienco@di.unito.it

This paper is a substantial extension of the preliminary work published in [Ienco et al. 2009].

that take values in a (usually) finite set of unordered nominal values. This makes impossible even to rank or compute differences between two values of the feature vectors. For categorical data the simplest comparison measure is *overlap* [Kasif et al. 1998]. The proximity between two multivariate categorical entities is proportional to the number of attributes in which they match. Other metrics, such as the Jaccard coefficient, are derived from *overlap* and have been adopted in several (partitional and hierarchical) clustering algorithms [Huang 1998; Guha et al. 1999; Andritsos et al. 2004].

Clearly, these distance metrics do not distinguish between the different values taken by the attribute, since they only measure the equality between pair of values. This is a strong limitation for a clustering algorithm, since it prevents to capture similarities that are clearly identified by human experts. For instance, given an attribute like *City*, which takes values in the set $\{Paris, Rome, Florence\}$ it is obvious that Florence is more similar to Rome than to Paris, from a geographic point of view. However, in some other contexts, Paris might be more similar to Rome, since both of them are capitals, and they may share similar behaviors.

In literature some measures that take into consideration the context of the features, have also been employed but refer to continuous data, e.g., Mahalanobis distance.

In this paper we present a new methodology to compute a context-based distance between values of a categorical variable and apply this technique to clustering categorical data with a hierarchical technique. For the introduction of our technique, consider the dataset described in figure 1(a), representing the set *Person*. It has two categorical attributes: *City{Milan, Turin, Florence}* and *Sex{Male,Female}*. The contingency table in Figure 1(b) shows how these values are distributed in the persons of the dataset. We observe that *City=Florence* occurs only with *Sex=Female* and *City=Turin* occurs only with *Sex=Male*. Conversely, *City=Milan* is satisfied both when *Sex=Male* and *Sex=Female*. From this distribution of data, we infer that, in this particular context, *Florence* is more similar to *Milan* than to *Turin* because the probability to observe a person of a given sex is closer.

Sex	**City**
Male	Turin
Female	Milan
Male	Turin
Male	Milan
Female	Florence

(a)

	Turin	Milan	Florence
Female	0	1	1
Male	2	1	0

(b)

Fig. 1. *Person*: a sample dataset with categorical attributes (a) and its related contingency table (b).

From this example we can deduce that the distribution of the co-occurrence table may help to define a distance between values of a categorical attribute. To this purpose, we propose a two-step method described in the following.

Let us denote by $F = \{X_1, X_2, \ldots, X_m\}$ the set of m attributes describing the dataset instances. Let us denote by $Y \in F$ the target, a categorical attribute on whose domain values we want to learn the distances.

(1) For Y, first identify a suitable context constituted by a subset of the attributes in F composed of $X_i \neq Y$, such that each attribute X_i belonging to the context is somehow

"correlated" to the attribute Y. This notion, that we will implement by means of the use of mutual information between Y and the context attributes, helps in deriving a set of target related attributes.

(2) In a second time, employing the distribution of the values of Y and each of the context attributes, we derive the measure of distance between the values of Y. For any pair of values (y_i, y_j) of Y, we measure the distance between the distributions of y_i and y_j in objects having the same values for the context attributes.

Concerning the first point, in this paper, we focus on data-driven methods for selecting a good context for a given attribute. However, an analyst could perform this context selection manually, exploiting its knowledge about the domain. In a knowledge-driven approach, for each given attribute, the analyst could select a subset of attributes of interest, following its knowledge base. We come back to the first example reported beforehand: when we consider the distance between two values of the *City* attribute, an analyst could decide to compute such a distance using, as context, some income-related features. Another possibility is to compute distances on a context based on health characteristics (if any). These two contexts would probably give rise to two different distance measures for the values of the *City* attribute. This difference might not be such evident when inferring the context directly from the data. Although these points deserve to be discussed further, in this paper we do not consider a user-driven context selection framework, since its validation would be rather subjective and application-dependent. Instead, here, we focus on a fully unsupervised framework, which, in our opinion, allows to produce fair experimental comparisons with other approaches, without loss of generality.

The key contributions of our work are the following:

—we introduce a new method, called *DILCA*, to compute the distance between any pair of values of a specific categorical attribute; notice that this distance-learning approach is independent of any subsequent learning on the actual instances (e.g., nearest-neighbors classification or distance-based clustering);

—we provide two methods for the selection of a suitable context: (i) a parametric method, and (ii) a fully automatic one;

—we show the impact of *DILCA* within an agglomerative hierarchical clustering algorithm and compare it with other three groups of algorithms from the state of the art; we evaluate results by means of three evaluation measures;

—we study the scalability of *DILCA* with respect to both the number of instances and the number of attributes: we show that *DILCA* can manage thousands of categorical attributes;

—we provide a study on the computational complexity of our solution;

—we provide an empirical study in which we break down the computational time of a usual clustering algorithm in all the times required by its successive steps. We show that the distance learning time due to *DILCA* constitutes the lowest fraction on the overall total.

The remainder of this paper is organized as follows: Section 2 briefly explores the state of the art in categorical data clustering. The theoretic fundamental details are presented in Section 3 while the technical issues of our distance learning approach are provided in

Section 4. In Section 5 we present the results of a comprehensive set of experiments on low and high-dimensional real-world and synthetic data, as well as a scalability analysis. Finally, Section 6 concludes.

2. RELATED WORK

Clustering is an important task in data mining, in information retrieval and in a wide range of analytical and scientific applications [Han and Kamber 2000]. The goal of clustering is to find a partition of the instances by optimization of a predefined distance measure or an objective function. The problem is particularly difficult when categorical attributes are involved in the clustering process because a definition of the distance between the values of a categorical attribute is not immediately available. In literature, many approaches to categorical data clustering have been proposed. Most of them try to optimize a global objective function without using any notion of distance between the values of the categorical attribute. Furthermore they suffer in terms of efficiency and time complexity with large data sets.

One of the first works in the field of categorical clustering is K-*MODES* [Huang 1998]. It extends K-*Means* algorithm for categorical data. A cluster is represented as a single instance, or data point, in which each attribute assumes the most frequent value in the database. Therefore, in K-*MODES* the similarity of an unlabeled data point and a cluster representative can be simply computed by the overlap distance [Kasif et al. 1998].

Another approach to categorical clustering is *ROCK* [Guha et al. 1999]. It employs links between pairs of data points in order to measure their similarity/proximity. An instance belongs to the neighborhood of another instance if the Jaccard similarity between them exceeds a user-defined threshold. It heuristically optimizes a cluster quality function with respect to the number of links between the cluster members. The basic algorithm is hierarchical and has cubic complexity in the size of the data set, which makes it unsuitable for large datasets.

CACTUS [Ganti et al. 1999] is a combinatorial search-based algorithm employing summary information of the dataset. $CACTUS$ first computes the projections of the data points onto the individual attributes. Then the projections are combined to form clusters over multiple attributes which are validated against the original dataset. In the validation phase if the support (cardinality) of some discovered cluster is less than a user-specified threshold, the cluster is removed. This approach optimizes a local function of the partition obtained by the clustering process.

Same approaches based on information theory have been proposed in literature. *COOLCAT* [Barbara et al. 2002] is based on the idea of entropy reduction within the generated clusters. It tries to optimize a global function based on entropy. It first bootstraps itself using a sample of maximally dissimilar points from the dataset to create initial clusters. The remaining points are then added incrementally. Naturally, this approach is highly dependent on the order of point selection. To mitigate this dependency, the authors propose to remove the "worst fitting" points at defined times during the execution and cluster them again. Li et al. [2004] propose an iterative algorithm to find optimal data partitions that minimize an entropy-based criterion. In this approach the authors employ a Monte Carlo process to randomly swap instances between clusters. Updates are retained when entropy decreases. The algorithm iterates the process until clusters remain unchanged.

LIMBO [Andritsos et al. 2004], is a scalable hierarchical categorical clustering algo-

rithm built on the Information Bottleneck framework. As a hierarchical algorithm, *LIMBO* is not as fast as partitional methods. The algorithm builds Distributional Cluster Features (DCF) trees to summarize the data in k clusters. Each node in a DCF tree contains statistics on a subset of instances. Starting from DCFs and the number of clusters k, a scan over the whole data set is performed to assign each instance to the cluster with the closest DCF. In the article the authors introduce an approach to define a distance measure for categorical tuples using the IB framework, but they do not experiment their intuition.

CLICKS [Zaki and Peters 2005] is a clustering algorithm based on graph/hyper graph partitioning. In general the cost of clustering with graph structures is acceptable, provided that the underlying data is low dimensional. *CLICKS* finds clusters in categorical datasets based on a search method for k-partite maximal cliques. The vertexes of the graph represent the value of the different attributes. In the graph there is an edge between two vertexes if the two attribute-values occur in the same instance. All maximal k-partite cliques in the graph are enumerated and the support of the candidate cliques within the original dataset is verified to form the final clusters. The crucial step of the algorithm is the computation of the strongly connected components, pairs of attribute values whose co-occurrence is above a specified threshold.

[Boriah et al. 2008] present an evaluation of different similarity measure for categorical data. The authors use more then ten measures for categorical data and they experiment their performance using the anomaly detection task. All of these measures use different heuristic to weigh the mismatch of the values of the same attribute. Differently from our approach, all these measures use only the distribution of the single attribute to compute similarity/distance between different values of the same attribute. Our approach, instead, uses the cross-information between attributes to weigh the mismatch. This type of information is extracted using the notion of context that we explain in the following section.

Another notable attempt of computing a distance for categorical attributes is [Ahmad and Dey 2007] whose authors propose a probabilistic framework which considers the distribution of all the attributes in the dataset. To compute the distance between two values y_i and y_j of a categorical attribute $Y \in F$, they propose to partition the set of remaining attributes into two sets W and $\tilde{W}$ ($W \cup \tilde{W} = F \setminus Y$) such that W is related to y_i, and $\tilde{W}$ is related to y_j. The retained partitioning is the one that optimizes $p(y_i|W) - p(y_j|\tilde{W})$. Then, they consider $p(y_i|W) - p(y_j|\tilde{W})$ as the distance value between y_i and y_j. In [Ahmad and Dey 2007], the authors only compare their approach with K-*MODES*. Moreover, since they consider distributions for all attributes, this method is not suitable for noisy and high-dimensional datasets.

Notice that learning distances between categorical attributes in supervised settings is quite an old task in machine learning. For instance, in [Stanfill and Waltz 1986], the authors propose a different approach to compute distances between symbols for a prediction task. In particular, they propose a "value difference metric" that takes into account how two values of an attribute are distributed w.r.t. a goal feature Y in the following way:

$$vdm(x_i, x_j) = \sqrt{\sum_{y_k \in Y} P(y_k|x_i)^2} \sum_{y_k \in Y} |P(y_k|x_i) - P(y_k|x_j)|^2$$

where $P(y_k|x_i)$ is the probability of class y_k given that attribute X has the value x_i. This

metric only considers how an attribute is distributed w.r.t. a class variable. Thus, it is only suitable for supervised problems. Moreover, the computed distance is asymmetric, i.e., $vdm(x_i, x_j) \neq vdm(x_j, x_i)$.

3. THE $DILCA$ APPROACH

In this section we present $DILCA$ (DIstance Learning for Categorical Attributes) for computing distances between any pair of values of a categorical attribute.

Let us consider the set $F = \{X_1, X_2, \ldots, X_m\}$ of m categorical attributes. Let $D = \{d_1, d_2, \ldots, d_n\}$ be the dataset of instances defined over F.

We denote by a lower case letter $x_j \in X_i$ a specific value of an attribute X_i. We refer to the cardinality of an attribute (also referred to as a feature) X_i as $|X_i|$.

We denote by Y the target attribute, which is a specific attribute in F that constitutes the target of the method, that is, on whose values we need to compute the distances. During the discussion if we need to distinguish an attribute in F from another one in the context we adopt the notation X_i. Otherwise, if we need to refer to a generic context attribute, distinguishing it with respect to the target attribute Y, we use simply X.

From the example in Section 1 it turns out that the distribution of values of an attribute X can be informative about the way in which another attribute Y is distributed in the dataset objects. Thanks to our $DILCA$ framework we can infer a context-based distance between any pair of values (y_i, y_j) of the target attribute Y on the basis of the similarity between the probability distributions of y_i and y_j given the context attributes, called $context(Y) \subseteq F \setminus Y$.

The core part of our approach consists in computing the distance between each pair of values of the target attribute Y. To compute the distance between y_i and y_j where $y_i \in Y$, $y_j \in Y$ we use the following formula:

$$d(y_i, y_j) = \sqrt{\frac{\sum_{X \in context(Y)} \sum_{x_k \in X} (P(y_i|x_k) - P(y_j|x_k))^2}{\sum_{X \in context(Y)} |X|}} \quad (1)$$

For each context attribute X_i we compute the conditional probability for both the values y_i and y_j given the values $x_k \in X_i$ and then we apply the Euclidean distance. The Euclidean distance is normalized by the total number of considered values. We obtain the total number of values by $\sum_{X \in context(Y)} |X|$.

Our distance measure is an application of the Euclidean distance. As such, our definition of distance is a metric. With our formulation we can see that our distance is bounded between 0 and 1: $0 \leq d(y_i, y_j) \leq 1$.

This definition of distance strongly depends on the context associated to each attribute. How to choose this context is then a crucial point in our approach, since a wrong selection would lead to weak clustering results. As we anticipated in Section 1, the analyst is free to determine an application-specific context for each attribute, exploiting her/his knowledge on the domain. Nevertheless, in the following, we provide two automatic data-driven methods that, for each given attribute, determine a suitable context.

3.1 Context selection

In real applications there could be several attributes that would make complex the context. In order to simplify the determination of the context, we investigate the problem

of selecting a good (informative) set of features w.r.t. a given one. This is a classic problem in data mining named feature selection. Feature selection is a preprocessing step of data mining. Its goal is to select a subset of relevant and not redundant features and discard all the other ones w.r.t. a given class attribute (supervised feature selection [Guyon and Elisseeff 2003]). In this branch of research many approaches for measuring the correlation/association between two features have been proposed. An interesting metric is the *Symmetric Uncertainty*, introduced in [Yu and Liu 2003]. This measure is an association-based measure inspired by information theory. Symmetric Uncertainty is derived from entropy: it is a measure of the uncertainty of a random variable. The entropy of a random variable Y is defined as:

$$H(Y) = -\sum_i P(y_i) \log_2(P(y_i))$$

where $P(y_i)$ is the probability of the value y_i of Y. The entropy of Y after having observed the values of another variable X is defined as:

$$H(Y|X) = -\sum_j P(x_j) \sum_i P(y_i|x_i) \log_2(P(y_i|x_i))$$

where $P(y_i|x_i)$ is the probability that $Y = y_i$ after we have observed that $X = x_i$. The information about Y provided by X is given by the *information gain* [Quinlan 1993] which is defined as follows:

$$IG(Y|X) = H(Y) - H(Y|X)$$

When $IG(Y|X_i) > IG(Y|X_j)$ then the feature X_i is more correlated to Y than X_j. Moreover, the Information gain is symmetrical for two random variables X and Y [Yu and Liu 2003].

The Symmetrical Uncertainty is then defined as follows:

$$SU(Y, X) = 2 \cdot \frac{IG(Y|X)}{H(Y) + H(X)} \tag{2}$$

It compensates for information gain's bias toward attributes with more values and normalizes its values to the range [0,1] (1 indicates that knowledge of the value of either Y or X completely predicts the value of the other variable; 0 indicates that Y and X are independent).

During the step of context selection, we select a set of context attributes for a given target attribute Y. This context, named $context(Y)$, is such that the attributes X_i belonging to this set have a high value of $SU(Y, X_i)$. Determining an adequate number of attributes for $context(Y)$ is not trivial.

Given a target attribute Y, we compute $SU(Y, X_i)$ for each attribute $X_i \neq Y$. We denote this Symmetric Uncertainty by $SU_Y(X_i)$.

So far, we have considered information theory measures for categorical attributes only. However, we can apply these metrics to the case of mixed categorical and continuous attributes using *differential entropy* [Verdugo and Rathie 1978], which extends the idea of entropy to continuous probability distributions:

$$H(X) = -\int_{\mathbb{X}} f(x) \log f(x) dx$$

where $\mathbb{X}$ is the domain (support) of X and $f(x)$ is the probability density function of X. When f comes from a standard probability distribution (e.g., normal, uniform, exponential) the entropy can be computed directly [Verdugo and Rathie 1978]. Otherwise, it is possible to discretize the continuous attributes and use the metrics described beforehand for the case of categorical attributes.

In this work we propose two alternative ways to adopt the Symmetric Uncertainty for the selection of an adequate context. The first one is parametric and employs a heuristic based on the mean value of SU for a specific target attribute Y. This solution is suitable when a user has enough information about the distribution of the attribute values inside the dataset. The second one is non parametric. This solution uses a well-known strategy in the feature selection field. This strategy allows to compute a good $context(Y)$ for each attribute and eliminates loosely informative attributes (called redundant) inside the specific $context(Y)$. In detail, the two strategies are performed as follows:

(1) The first heuristic is based on the mean value of SU for a specific target attribute Y. The mean of this quantity is:

$$mean(SU_Y) = \frac{\sum_{X_i \in F \setminus Y} SU_Y(X_i)}{|F \setminus Y|} \tag{3}$$

For the determination of the context of the target attribute Y we use the attributes that satisfy the following inequality:

$$context(Y) = \{X_i \in F | X_i \neq Y \ \ \wedge \ \ SU_Y(X_i) \geq \sigma \cdot mean(SU_Y)\}$$

where $\sigma \in [0, 1]$ is a trade-off parameter that controls the influence of the mean value. In the paper we refer to this approach with $DILCA_M$.
According to this heuristic, at least one attribute is assigned to $context(Y)$. This is simple to demonstrate. If $SU_Y(X_i)$ is the same for all X_i then $SU_Y(X_i) = mean(SU_Y)$ for all X_i; in this case all X_i would be selected in $context(Y)$. Otherwise, there exists at least one attribute X_i such that $SU_Y(X_i) \geq mean(SU_Y)$. Then X_i would be selected.

(2) The second approach is based on the $FCBF$ algorithm for feature selection proposed in [Yu and Liu 2003]. In this non-parametric approach we consider two issues: the *r*elevance and the *r*edundancy between attributes.
Relevance:. For the target attribute Y, we measure the relevance of all the attributes $X_i \neq Y$ in F. An attribute X_1 is more relevant than a second attribute X_2 with respect to the target Y if $SU_Y(X_1) \geq SU_Y(X_2)$. We rank this set of attributes according to decreasing values of $SU_Y(X_i)$. Therefore, the most relevant features are at the top of the ranking.
Redundancy:. After the ranking step we remove all those attributes that are redundant. A feature X_j is redundant with respect to X_i if X_i is more relevant with respect to the target than X_j (according to the definition of relevance) and $SU_{X_i}(X_j) > SU_Y(X_j)$. As a consequence, X_j can be safely removed from the ranked list of attributes.
With these two steps we obtain a $context(Y)$ that automatically contains relevant and not redundant attributes w.r.t. Y attribute. In the paper we refer to this approach with the name $DILCA_{RR}$ (the double 'R' stands for relevance and redundancy).

4. $DILCA$ IMPLEMENTATION

In this section we introduce the algorithmic details for $DILCA$. Our approach is based on two steps:

(1) **Context selection:** selection of a relevant subset of the whole attributes set F that we use as the context for the determination of the distances on the values of a given attribute Y;
(2) **Distance computation:** computation of the distance measure between pair of values of Y using the context defined in the previous step.

For the first step, we describe the two solutions given in the previous section, and we discuss the complexity of the two approaches for the problem of learning a good context for a target variable.

In Algorithm 1 we show the adopted procedure for the computation of the correlation matrix between each pair of attributes based on Symmetric Uncertainty. This algorithm takes as parameter the entire data set D. Then, function $ComputeCO(D, X, Y)$ scans the dataset D once and for each instance of D it updates the contingency tables (CO_{X_iY}) of each pair of attributes (X_i, Y), where Y is any categorical attribute in F, taken as the target, and X_i any other attribute in F. These tables are used to compute the Symmetric Uncertainty between attributes to be stored in $matrixSU$.

Each column of $matrixSU$ is a vector that contains the Symmetrical Uncertainty between a certain attribute X_i and all the other ones $X_j \in F$. A particular column of $matrixSU$ regards the symmetrical uncertainty of the target Y. Let us denote the column in $matrixSU$ that regards the Symmetrical Uncertainty of attribute Y as $VectorSU_Y$. (Similarly, for any other attribute $X \in F$, we have $VectorSU_X$). Each attribute $X \in F$ is used as an index of $VectorSU_Y$. $VectorSU_Y$ at the position identified by X contains $SU_Y(X)$. We denote it by $VectorSU_Y[X]$.

4.1 $DILCA_M$

In Algorithm 2 we propose the parametric approach in which $context(Y)$ is built. At first, it computes the mean of the vector $VectorSU_Y$ which constitutes a column of $matrixSU$. Then, at lines 4 and 5, it selects the attributes that will be included in the context of Y. When the features are chosen, ***DistanceComputation*** function (see successive Algorithm 4) computes the distance matrix between each pair of values of the attribute Y by application of equation (1).

4.2 $DILCA_{RR}$

In Algorithm 3 we extend to an unsupervised setting the $FCBF$ approach proposed in [Yu and Liu 2003]. $FCBF$ selects a relevant and non redundant set of attributes for a given attribute Y. In the supervised approach the target Y is the class attribute. $FCBF$ uses some heuristic about attributes. It assumes that if two attributes X_i and X_j are relevant and one of them is found to be redundant then it is removed.

For the context selection step of $DILCA_{RR}$ (see Algorithm 3) we implement the idea just explained as follows. Given that both X_i and X_j are relevant to the target, one of them is considered redundant if the Symmetrical Uncertainty that links them is higher than the Symmetrical Uncertainty that links each of them to the target. In other terms, if both the

following conditions are satisfied:

(1) $SU_{X_j}(X_i) > SU_Y(X_i)$
(2) $SU_{X_j}(X_i) > SU_Y(X_j)$

In particular, X_j is removed if X_i is more relevent to the target Y. In terms of SU it is: $SU_Y(X_j) < SU_Y(X_i)$.

Then, after the context of Y is formed, we compute the distance matrix on the values of Y similarly to $DILCA_M$, by *DistanceComputation* function.

Algorithm 1 computeCorrelationMatrix(D)

for all $X, Y \in F | X \neq Y$ **do**
  $CO_{XY} = ComputeCO(D, X, Y)$
  $matrixSU[X][Y] = SU(CO_{XY})$
end for
return $matrixSU$

Algorithm 2 $DILCA_M$($VectorSU_Y$, Y, σ)

$mean$ = computeMean($VectorSU_Y$)
$context(Y) = \emptyset$
for all $X \in F$ **do**
  if $VectorSU_Y[X] \geq \sigma \cdot mean$ **then**
    insert(X,$context(Y)$)
  end if
end for
$DistMatrix_Y$ = DistanceComputation(Y, $context(Y)$)
return $DistMatrix_Y$

Algorithm 3 $DILCA_{RR}$($matrixSU$, Y)

$VectorSU_Y$ = $MatrixSU[Y]$
$context(Y) = \{X \in F | X \neq Y\}$
sort $VectorSU_Y$ in descending order // RELEVANCE step
for all $VectorSU_Y[X]$ starting from the top position and s.t. $X \neq Y$ **do**
  $VectorSU_X$ = $MatrixSU[X]$
  for all $VectorSU_Y[K]$ s.t. $VectorSU_Y[K] \leq VectorSU_Y[X]$ **do**
    – REDUNDANCY step
    if ($VectorSU_X[K] \geq VectorSU_Y[K]$) **then**
      erase attribute K from $context(Y)$
      erase $VectorSU_Y[K]$
    end if
  end for
end for
$DistMatrix_Y$ = DistanceComputation(Y, $context(Y)$)
return $DistMatrix_Y$

Algorithm 4 DistanceComputation(Y, $context(Y)$)

1: **for all** $y_i, y_j \in Y | y_i \neq y_j$ **do**

2: $DistanceMatrix[y_i][y_j] = \sqrt{\frac{\sum_{X \in context(Y)} \sum_{x_k \in X} (P(y_i|x_k) - P(y_j|x_k))^2}{\sum_{X \in context(Y)} |X|}}$

3: **end for**

4: **return** $DistanceMatrix_Y$

4.3 Complexity

Before computing the distance matrix on the target values, we must store the co-occurrence of the values of any pair of attributes (X, Y). This needs the computation of l matrices $CO_{X,Y}$, with $l = m * (m-1)/2$ the number of all the pairs of attributes in F and m the total number of attributes in F. In order to build these matrices we need to perform a complete scan of the entire data set (which has a complexity of n). Later, we will use the co-occurrence matrices $CO_{X,Y}$ also to compute $matrixSU$. $matrixSU$ is $m \times m$. Follows the complexity of the construction of the distance matrix in the two approaches $DILCA_M$ and $DILCA_{RR}$:

—$DILCA_M$: from $matrixSU$ we compute $mean(SU_Y)$ and then select the right context making use of σ. For the computation of the distance matrix on the values of each target attribute we can use the co-occurrence matrices $CO_{X,Y}$ without the necessity of further scans of the dataset. From this, we derive that our algorithm only needs to scan the dataset once. In conclusion, $DILCA_M$ has complexity $O(nm^2)$, with n the number of instances and m the number of attributes.

—$DILCA_{RR}$: Using $matrixSU$, for each attribute we select the context with the algorithm 3. From the study in [Yu and Liu 2003] the authors show that the complexity of this approach is $O(m \log m)$. In our case we perform this analysis for each attribute and we see that the complexity becomes $O(m^2 \log m)$. Again, to compute the distance matrix for each attribute we can use the co-occurrence matrices $CO_{X,Y}$ without the necessity of further scans of the dataset. From this analysis we can infer that the complexity of $DILCA_{RR}$ is $O(nm^2 \log m)$.

5. EXPERIMENTS AND RESULTS

In this section we present a comprehensive evaluation of our approach. We coupled $DILCA$ with a standard hierarchical clustering method. We used Ward's hierarchical clustering algorithm. It uses as input a matrix with the distances between each pair of objects in the dataset. In order to compute this matrix we must define the distance between two objects. In the definition of the distance between two objects, we can combine the distances between the pairs of values of any attribute in the two objects. To obtain the distance between each pair of values of the categorical attributes we apply $DILCA_M$ or $DILCA_{RR}$. From a procedural point of view, each categorical attribute is taken as the target attribute and each of the remaining attributes forms a candidate for the target context. At this point we obtain a set of attribute distance matrices that we denote by $\{distMatrix_{X_i}\}$. Since our method provides the distance between pairs of values of categorical attributes, this distance can be integrated in the standard distance measures (like the euclidean one) in an

m-dimensional space as follows:

$$distObj(o_k, o_j) = \sqrt{\sum_{X_i \in F} distMatrix_{X_i}(o_k[X_i], o_j[X_i])^2} \quad (4)$$

where $distObj(o_k, o_j)$ is the distance between the two objects o_k and o_j; $o_k[X_i]$ and $o_j[X_i]$ are the values of the attribute X_i in objects o_k and o_j respectively; $distMatrix_{X_i}(o_k[X_i], o_j[X_i])$ is the distance obtained using $DILCA_M$ or $DILCA_{RR}$ between the two values $o_k[X_i]$ and $o_j[X_i]$.

5.1 Competitors

We compare our approaches with three different groups of competitors and we evaluate the obtained results using three objective evaluation measures described in the Section 5.2.

5.1.1 *Comparison using Base-Line methods.* We perform a first set of experiments comparing our approaches with two base-line methods.

The first base-line is a distance based on the *Jaccard* similarity. It is a normalized version of the *overlap* [Kasif et al. 1998]. Given two instances, we compute the *Jaccard* distance as:

$$dist_{Jaccard}(o_k, o_j) = 1 - \frac{\sum_{X_i \in F} 1(o_k[X_i] = o_j[X_i])}{2 \cdot |F| - \sum_{X_i \in F} 1(o_k[X_i] = o_j[X_i])}$$

where $1(\cdots)$ is the indicator function.

The second base-line that we employ is the *Pearson* distance. This measure is derived from the Pearson's coefficient [Han and Kamber 2000]. Given two variables, the Pearson's correlation coefficient is defined as the covariance of the two variables divided by the product of their standard deviations. The Pearson's correlation coefficient ranges between -1 and 1. It also uses negative values to indicate that the two variables are anti-correlated. Given two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, the Person's coefficient is given by:

$$\rho = \frac{\sum_{i=1}^{n}(a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^{n}(a_i - \bar{a})^2}\sqrt{\sum_{i=1}^{n}(b_i - \bar{b})^2}}$$

where $\bar{a}$ and $\bar{b}$ are the vector means. We define the *Pearson* distance as $1 - \rho$. It takes values between 0 and 2. Since the *Pearson* distance is designed for numerical vectors, we use a *DILCA*-like setting to adapt this metric to categorical attributes. Given a target attribute Y, for each value $y_i \in Y$ we build a vector whose components are the co-occurrences of y_i with all the values of the attributes in $\{F \setminus Y\}$. Using this representation we compute the *Pearson* distance between each pair of values of the attribute Y. We repeat this process for each attribute. At the end we obtain a set of distance matrices and we employ Equation 4 to compute the distance between two instances. We coupled both base-line approaches with Ward's clustering and we refer to the first base-line method as *JACCARD* and to the second base-line method as *PEARSON*.

5.1.2 *Comparison using Categorical Clustering.* In this second bunch of experiments we want to compare the performance of our approaches w.r.t. state-of-the-art method for clustering categorical data. To do this we compare *DILCA*$_M$ and *DILCA*$_{RR}$

with *ROCK* [Guha et al. 1999] and *LIMBO* [Andritsos et al. 2004]. We also implemented the distance learning method described in [Ahmad and Dey 2007] within Ward's clustering algorithm adopted in *DILCA*: we refer to this solution as *DELTA*. Also for the *DELTA* approach we use Equation 4 to compute the distance between two objects.

5.1.3 *Comparison using Similarity measures for categorical data.* For the last set of experiments, we implemented three of the measures proposed in [Boriah et al. 2008] (please, refer to this paper for further details). In particular we use the *LIN*, *OF* (Occurrence Frequency) and *Goodall3* measures that are shown to outperform the other discussed similarity/distance measures. The *LIN* measure is derived from Information Theory, and it is related to compression. The *OF* measure assigns high similarity when the mismatch involves two frequent values (and low similarity for values that are less frequent). The *Goodall3* measure assigns high similarities if the matching values are infrequent regardless of the frequencies of the other values. All these three measures are similarity measures. Hence, to transform them in distance we follow the suggestion given in [Boriah et al. 2008]. Given a similarity measure sim we obtain the distance transformation using the following strategy:

$$dist = \frac{1}{1 + sim}$$

We can adapt each of these measures to produce a set of matrices (like *DILCA*$_M$ and *DILCA*$_{RR}$). As for the previous groups of competitors, we compute the distance between two objects using Equation 4 and we couple them with Ward's clustering algorithm. In the experiments, we refer to these three methods as *LIN*, *OF* and *GOODALL3*.

5.2 Clustering Evaluation Metrics

The definition of the clustering quality is often a hard and subjective task. Therefore, we use three objective criteria to evaluate the results: Purity, Normalized Mutual Information and Adjusted Rand Index. These methods make use of the correspondence between the original class information of each object and the cluster to which the same objects have been assigned.

We denote by $\mathbf{C} = \{C_1 \dots C_J\}$ the partition built by the clustering algorithm on objects, and by $\mathbf{P} = \{P_1 \dots P_I\}$ the partition inferred by the original classification. J and I are respectively the number of clusters $|\mathbf{C}|$ and the number of classes $|\mathbf{P}|$. We denote by n the total number of objects.

The first measure of the quality of a clustering solution is *purity* in terms of the class which the cluster objects belong to. In order to compute purity each cluster is assigned to the majority class of the objects in the cluster. Then, the accuracy of this assignment is measured by counting the number of correctly assigned objects divided by the total number of objects n:

$$\textbf{Purity}(\mathbf{C}, \mathbf{P}) = \frac{1}{n} \sum_j max_i |C_j \cap P_i| \tag{5}$$

Notice that Purity is sensible to the presence of imbalanced classes.

The second metric provides an information that is independent from the number of clusters [Strehl et al. 2002]. This measure takes its maximum value when the clustering partition matches completely the original partition. We can consider NMI as an indicator of

the purity of the clustering result. NMI is computed as the average mutual information between any pair of clusters and classes:

$$\mathbf{NMI} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} x_{ij}\log\frac{n*n_{ij}}{n_i n_j}}{\sqrt{\sum_{i=1}^{I} n_i \log\frac{n_i}{n}\sum_{j=1}^{J} n_j \log\frac{n_j}{n}}} \tag{6}$$

where n_{ij} is the cardinality of the set of objects that occur both in cluster i and in class j; n_i is the number of objects in cluster i; n_j is the number of objects in class j; n is the total number of objects. I and J are respectively the number of clusters and the number of classes.

The third metric is the adjusted Rand index [Hubert and Arabie 1985]. Let a be the number of object pairs belonging to the same cluster in $\mathbf{C}$ and to the same class in $\mathbf{P}$. This metric captures the deviation of a from its expected value corresponding to the hypothetical value of a obtained when $\mathbf{C}$ and $\mathbf{P}$ are two random, independent partitions. The expected value of a denoted by $E[a]$ is computed as follows:

$$E[a] = \frac{\pi(C)\cdot\pi(P)}{n(n-1)/2}$$

where $\pi(C)$ and $\pi(P)$ denote respectively the number of object pairs from the same clusters in $\mathbf{C}$ and from the same class in $\mathbf{P}$. The maximum value for a is defined as:

$$max(a) = \frac{1}{2}\left(\pi(C)+\pi(P)\right)$$

The agreement between $\mathbf{C}$ and $\mathbf{P}$ can be estimated by the adjusted rand index as follows:

$$ARI(\mathbf{C},\mathbf{P}) = \frac{a - E[a]}{max(a) - E[a]} \tag{7}$$

Notice that this index can take negative values, and when $AR(\mathbf{C},\mathbf{P}) = 1$, we have identical partitions.

5.3 Datasets for Categorical Clustering Evaluation

For the evaluation of our distance learning approach on categorical data, we used two collections of datasets.

The first collection consists in real world data sets downloaded from the UCI Machine Learning Repository [Blake and Merz 1998].

The second collection contains synthetic datasets produced by a data generator [Melli 2008], which employs a Gaussian distribution to generate the data.

The main characteristics of the datasets are summarized in Figure 2. Notice that some datasets contains numerical variables. In order to perform experiments on it and compare *DILCA* with some of the competitors that are able to treat only categorical attributes, we had to discretize the continuous attributes using the supervised method proposed in [Witten and Frank 2005].

5.4 Experimental Settings

In this section, we report on the performance results of our proposal, that consists in the two approaches, *DILCA*$_M$ and *DILCA*$_{RR}$ coupled with Ward hierarchical clustering.

Dataset	Type	Instances	Features	Values	Classes
Audiology	Real	226	59	154	24
Votes	Real	435	16	32	2
Mushroom	Real	8124	22	117	2
Car	Real	1728	6	21	4
Dermatology	Real	336	34	131	6
Soybean	Real	683	35	100	19
Adult	Real	4884	14	120	2
Post-operative	Real	90	8	25	3
Titanic	Real	2201	3	10	2
Hepatitis	Real	155	19	38	2
Breast-cancer	Real	286	9	53	2
Ballons	Real	20	4	10	2
SynA	Synth	1000	50	1000	5
SynB	Synth	1000	50	2500	5
SynC	Synth	1000	100	2000	5
SynD	Synth	1000	100	5000	5

Fig. 2. Datasets characteristics

We run the experiments on a PC with a 2.66GHz Intel Core 2 Duo processor, 1024MB of RAM running Linux.

The experiments were conducted as follows: since the hierarchical algorithm returns a dendrogram which, at each level, contains a different number of clusters, we consider the level corresponding to the number of clusters equal to the number of classes. This is done for each method that employ the hierarchical Ward clustering. Moreover, for *DILCA*$_M$, we varied parameter σ between 0 to 1 with steps of 0.1, and we selected the value of σ which gave the best results. For *ROCK*, we set the threshold parameter between 0.2 to 1 with steps of 0.05. Also for this algorithm we retained the best obtained result. For *LIMBO*, we set ϕ parameter between 0 to 1 with steps of 0.25 and we report the best result obtained. This parameter influences the information loss during the merging phase.

5.5 Results

In the Tables 3, 4 and 5 we report the results of the comparative evaluation against *ROCK*, *LIMBO* and *DELTA*. When looking at the purity values, it may seem that *ROCK* achieves significantly better results than all other approaches for the following datasets: *Post-operative*, *Titanic*, *Hepatitis* and *Breast-cancer*. However, when we inspected the partitions returned by *ROCK*, we discovered that for these datasets, this algorithm generate some very small and pure clusters (containing few instances of the same class) and some huge clusters containing instances from different classes. Purity is biased by this unbalanced partitioning, but, if we look at NMI and ARI results, the values of these two metrics are below the values computed for our approaches in most cases. In this bunch of experiments, our approaches outperform the others in most cases (12 datasets), but, even when they fail in achieving the best results, at least one of the performance pa-

Dataset	DILCA$_M$	DILCA$_{RR}$	DELTA	ROCK	LIMBO
Audiology	38.05%	39.82%	35.84%	32.30%	**46.01**%
Vote	**91.95**%	89.43%	89.43%	83.90%	87.12%
Mushroom	**89.02**%	**89.02**%	**89.02**%	49.43%	88.95%
Soybean	68.08%	**71.74**%	71.60%	43.71%	55.64%
Dermatology	94.54%	**97.27**%	88.80%	84.97%	87.70%
Car	**70.08**%	**70.08**%	30.15%	70.08%	44.50%
Adult	68.59%	**76.16**%	64.95%	61%	68.22%
Post-operative	47.78%	35.56%	35.56%	**91.11**%	41.11%
Titanic	77.37%	60.84%	60.84%	**98.96**%	60.84%
Hepatitis	83.22%	69.67%	69.67%	**99.35**%	84.52%
Breast-Cancer	74.47%	74.47%	70.97%	**97.20**%	69.93%
Balloons	**100**%	**100**%	80%	90%	100%
SynA	99.2%	99.2%	73%	**100**%	87.6%
SynB	**93.7**%	93.5%	75.5%	56%	63.8%
SynC	**84.2**%	83.4%	34.79%	77.6%	48.7%
SynD	94.7%	92.8%	67.4%	**98.7**%	59.9%

Fig. 3. Purity results w.r.t. state-of-the-art categorical algorithms

Dataset	DILCA$_M$	DILCA$_{RR}$	DELTA	ROCK	LIMBO
Audiology	0.6006	**0.6040**	0.5604	0.3216	0.5122
Vote	**0.6009**	0.5278	0.4994	0.3446	0.4358
Mushroom	**0.5938**	**0.5938**	**0.5938**	0.05681	0.5684
Soybean	0.7902	0.7813	**0.8161**	0.6299	0.7458
Dermatology	0.9120	**0.9519**	0.8790	0.6372	0.8776
Car	0.0359	0.0359	**0.0621**	0.0359	0.0142
Adult	0.1272	0.0680	0.1894	0.0943	**0.2077**
Post-operative	**0.0674**	0.0289	0.0289	0.0527	0.0555
Titanic	**0.1673**	0.0235	0.0235	0.0357	0.0235
Hepatits	0.2403	0.1342	0.1342	0.0728	**0.2878**
Breast-Cancer	0.0741	0.0741	0.0765	**0.1252**	0.0957
Ballons	**1.0**	**1.0**	0.4325	0.2141	**1.0**
SynA	0.9790	0.9790	0.4427	**1.0**	0.7540
SynB	**0.8548**	0.8345	0.4167	0.3320	0.4458
SynC	**0.643**	0.6269	0.0215	0.4799	0.0916
SynD	**0.8705**	0.818	0.3652	0.0591	0.2337

Fig. 4. NMI results w.r.t. state-of-the-art categorical algorithms

rameters is close to the winning approach. On the other hand, the performance indexes achieved by *DILCA* are sensibly better than those obtained by the other approaches in many datasets (*Vote*, *Dermatology*, *Titanic*, *SynB*, *SynC* and *SynD*). Notice that the last three datasets are high-dimensional and significantly sparse (20 and 50 values per variable). This means that our approach provide accurate results also in these hard contexts. Moreover, we observed that *ROCK* is very sensitive to the parameter value. In these experiments we noticed that *DILCA*$_M$ obtains the best results when we set the σ parameter to values higher than 0.5. Indeed, during the distance learning phase on the categorical values, the algorithms consider as a context only small portions of the whole attribute set.

In the Tables 6, 7 and 8, instead, we report the results of the comparison of our approaches with *LIN*, *OF* and *GOODALL3*. Also in this case, when some of the other

Dataset	DILCA$_M$	DILCA$_{RR}$	DELTA	ROCK	LIMBO
Audiology	0.2103	0.2280	0.2104	0.0807	**0.2836**
Vote	**0.7031**	0.6207	0.6200	0.4563	0.5496
Mushroom	**0.6090**	**0.6090**	**0.6090**	-0.0011	0.6070
Soybean	0.5094	0.5109	**0.5635**	0.1734	0.4442
Dermatology	0.9158	**0.9542**	0.7234	0.4420	0.7179
Car	0.0129	0.0043	0.0043	0.0043	**0.0367**
Adult	0.1364	**0.1696**	0.0849	0.0453	0.1319
Post-operative	**0.0616**	-0.0128	-0.0128	0.0403	-0.0130
Titanic	**0.2744**	0.0002	0.0002	0.0186	0.0002
Hepatits	0.3903	0.1459	0.1459	0.0362	**0.4337**
Breast-Cancer	**0.159**	**0.159**	0.1533	0.0775	0.1504
Ballons	**1.0**	**1.0**	0.3262	0.1215	**1.0**
SynA	0.9803	0.9803	0.4554	**1.0**	0.7350
SynB	**0.8457**	0.8437	0.3563	0.2518	0.4016
SynC	**0.6498**	0.6302	0.0127	0.4631	0.0359
SynD	**0.8717**	0.8247	0.3589	0.0004	0.1825

Fig. 5. Adjusted Rand Index results w.r.t. state-of-the-art categorical algorithms

Dataset	DILCA$_M$	DILCA$_{RR}$	LIN	OF	GOODALL3
Audiology	38.05%	39.82%	**42.48%**	34.07%	41.14%
Vote	**91.95%**	89.43%	89.65%	86.90%	81.84%
Mushroom	**89.02%**	**89.02%**	**89.02%**	**89.02%**	**89.02%**
Soybean	68.08%	71.74%	72.18%	**75.70%**	70.13%
Dermatology	94.54%	**97.27%**	**97.27%**	84.42%	87.70%
Car	**70.08%**	**70.08%**	32.40%	41.67%	32.41%
Adult	68.59%	**76.16%**	62.63%	61.01%	61 %
Post-operative	**47.78%**	35.56%	40%	35.56%	37.78%
Titanic	**77.37%**	60.84%	60.84%	59.79%	59.79%
Hepatitis	**83.22%**	69.67%	80.64%	69.03%	71.61%
Breast-Cancer	**74.47%**	**74.47%**	68.53%	61.88%	56.64%
Balloons	**100%**	**100%**	80%	80%	80%
SynA	**99.2%**	**99.2%**	97.3%	86.7%	88.6%
SynB	**93.7%**	93.5%	90.2%	79.5%	82.8%
SynC	**84.2%**	83.4%	81.79%	52.8%	56.89%
SynD	**94.7%**	92.8%	96.3%	83.7%	83.4%

Fig. 6. Purity results w.r.t. advanced measures

approaches outperform *DILCA*, the performance indexes are in general close (except for *Car*). However, in many cases, when *DILCA* outperforms all other approaches, its performance parameters are much higher (e.g., in *Vote*, *Titanic*, *Ballons*, *SynA*, *SynB* for NMI, and in the same datasets, plus *Audiology*, *Adult* and *Post-operative* for ARI). We omit here the discussion on Purity, since its values are less significant than NMI and ARI, as shown beforehand. However, for completeness, we report them in Table 6.

Finally, we consider the last group of competitors, which employs two well known similarity coefficients (see Tables 9, 10 and 11). As expected, results for the standard Jaccard metrics are poor in general. The Pearson's coefficient, instead, in some cases achieves comparable results w.r.t. *DILCA*. In many cases, however, the NMI and ARI performance indexes are far from being comparable with those obtained by *DILCA*. This is the case for *Audiology*, *Vote*, *Dermatology*, *SynB*, *SynC* and *SynD* (when con-

Dataset	DILCA$_M$	DILCA$_{RR}$	LIN	OF	GOODALl3
Audiology	0.6006	**0.6040**	0.5536	0.4562	0.5535
Vote	**0.6009**	0.5278	0.5426	0.4226	0.3985
Mushroom	**0.5938**	**0.5938**	**0.5938**	**0.5938**	**0.5938**
Soybean	0.7902	0.7813	0.7988	**0.8138**	0.7845
Dermatology	0.9120	**0.9519**	0.9381	0.7333	0.8484
Car	0.0359	0.0359	0.0621	**0.0634**	0.0621
Adult	0.1272	0.0680	0.0295	0.1447	**0.1448**
Post-operative	**0.0674**	0.0289	0.0585	0.0380	0.0257
Titanic	**0.1673**	0.0235	0.0235	0.0168	0.0168
Hepatits	0.2403	0.1342	**0.2594**	0.2022	0.1816
Breast-Cancer	**0.0741**	**0.0741**	0.0599	0.0041	0.0227
Ballons	**1.0**	**1.0**	0.4325	0.4325	0.4325
SynA	**0.9790**	**0.9790**	0.9138	0.6702	0.7054
SynB	**0.8548**	0.8345	0.7751	0.593	0.6399
SynC	**0.643**	0.6269	0.595	0.2032	0.2582
SynD	0.8705	0.818	**0.882**	0.6188	0.612

Fig. 7. NMI results w.r.t. advanced measures

Dataset	DILCA$_M$	DILCA$_{RR}$	LIN	OF	GOODALl3
Audiology	0.2103	**0.2280**	0.1480	0.1025	0.1806
Vote	**0.7031**	0.6207	0.6280	0.5419	0.4040
Mushroom	**0.6090**	**0.6090**	**0.6090**	**0.6090**	**0.6090**
Soybean	0.5094	0.5109	**0.5577**	0.5408	0.5019
Dermatology	0.9158	**0.9542**	0.9507	0.6754	0.7023
Car	0.0129	0.0043	0.0129	**0.0544**	0.0129
Adult	0.1364	**0.1696**	0.0555	0.0398	0.0260
Post-operative	**0.0616**	-0.0128	0.0278	-0.0156	-0.0161
Titanic	**0.2744**	0.0002	0.002	-0.0009	-0.0009
Hepatits	**0.3903**	0.1459	0.3488	0.1411	0.1791
Breast-Cancer	**0.159**	**0.159**	0.12	0.0139	0.0142
Ballons	**1.0**	**1.0**	0.3262	0.3262	0.3262
SynA	0.9803	0.9803	0.9339	0.6937	0.7297
SynB	**0.8457**	0.8437	0.7798	0.5786	0.632
SynC	**0.6498**	0.6302	0.5986	0.1735	0.1798
SynD	0.8717	0.8247	**0.9092**	0.6301	0.6201

Fig. 8. Adjusted Rand Index results w.r.t. advanced measures

sidering NMI), and *Vote*, *Dermatology*, *Car*, *Adult*, *SynB* and *SynC* (when considering ARI).

5.5.1 *Impact of σ on* DILCA$_M$. We plot the behavior of *DILCA$_M$* to check how the σ parameter influences the algorithm.

We let vary the parameter σ from 0 to 1 with steps of 0.1. When the parameter is equal to 0 all the features are included in the context. We observed that the influence of different settings of σ on performance indexes is minimum on UCI datasets (see Figure 12 and 12). In most datasets, the variation in Purity and NMI is very low (the only exception is Vote). In Figure 12 we report only the sensitivity plot for those datasets in which the variation between the maximum value and the minimum one is greater or equal 0.05. We use the same policy for both Purity and NMI. For the purity this means

Dataset	DILCA$_M$	DILCA$_{RR}$	JACCARD	PEARSON
Audiology	38.05%	39.82%	45.13%	**48.67%**
Vote	**91.95%**	89.43%	86.2%	85.28%
Mushroom	**89.02%**	**89.02%**	**89.02%**	**89.02%**
Soybean	68.08%	71.74%	69.1%	**75.4%**
Dermatology	94.54%	**97.27%**	94.54%	87.43%
Car	70.08%	70.08%	32.4%	**99.82%**
Adult	68.59%	**76.16%**	58.87%	64.09%
Post-operative	47.78%	35.56%	40.0%	**54.44%**
Titanic	**77.37%**	60.84%	60.83%	77.32%
Hepatitis	**83.22%**	69.67%	74.83%	82.58%
Breast-Cancer	**74.47%**	**74.47%**	61.53%	**74.47%**
Balloons	**100%**	**100%**	80%	**100%**
SynA	99.2%	99.2%	93.0%	**100%**
SynB	**93.7%**	93.5%	87.2%	90.5%
SynC	**84.2%**	83.4%	67%	70.2%
SynD	**94.7%**	92.8%	91.5%	92.1%

Fig. 9. Purity results w.r.t. base-line measures

Dataset	DILCA$_M$	DILCA$_{RR}$	JACCARD	PEARSON
Audiology	0.6006	**0.6040**	0.6014	0.476
Vote	**0.6009**	0.5278	0.4039	0.4363
Mushroom	**0.5938**	**0.5938**	**0.5938**	**0.5938**
Soybean	**0.7902**	0.7813	0.7881	0.7731
Dermatology	0.9120	**0.9519**	0.8997	0.8089
Car	0.0359	0.0359	**0.0621**	0.0358
Adult	0.1272	0.0680	0.0811	**0.1293**
Post-operative	**0.0674**	0.0289	0.0211	0.055
Titanic	**0.1673**	0.0235	0.0234	**0.1673**
Hepatits	0.2403	0.1342	0.1775	**0.2423**
Breast-Cancer	**0.0741**	**0.0741**	0.0188	**0.0741**
Ballons	**1.0**	**1.0**	0.4325	**1.0**
SynA	0.9790	0.9790	0.8079	**1.0**
SynB	**0.8548**	0.8345	0.7126	0.7757
SynC	**0.643**	0.6269	0.3971	0.4286
SynD	**0.8705**	0.818	0.7587	0.7772

Fig. 10. NMI results w.r.t. base-line measures

that the variation is greater or equal to 5%.

On synthetic data, performance indexes grow with increasing values of σ (See Fig.13). Also in this figures, we report only those curves that show a significant variations.

Although there is no general law about how to choose this parameter, we estimate that, in general, its impact is less important than standard clustering parameters (such as, the number of clusters, or other algorithm-specific parameters).

5.5.2 *Scalability of DILCA$_{RR}$ and DILCA$_M$.* We introduce now a study on the scalability of our distance learning approach, coupled with hierarchical clustering algorithms. We evaluate the scalability varying the two dimensions of the dataset that may have an impact on time performances. The datasets are generated with [Melli 2008]. For *DILCA$_M$*,

Dataset	DILCA$_M$	DILCA$_{RR}$	JACCARD	PEARSON
Audiology	0.2103	0.2280	**0.2668**	0.1952
Vote	**0.7031**	0.6207	0.5218	0.4969
Mushroom	**0.6090**	**0.6090**	**0.6090**	**0.6090**
Soybean	0.5094	**0.5109**	0.4966	0.4539
Dermatology	0.9158	**0.9542**	0.9086	0.684
Car	**0.0129**	0.0043	**0.0129**	0.0042
Adult	0.1364	**0.1696**	0.0257	-0.0028
Post-operative	**0.0616**	-0.0128	-0.0218	0.0269
Titanic	**0.2744**	0.0002	0.0002	**0.2744**
Hepatits	**0.3903**	0.1459	0.2285	0.3801
Breast-Cancer	**0.159**	**0.159**	-0.0098	**0.159**
Ballons	**1.0**	**1.0**	0.3262	**1.0**
SynA	0.9803	0.9803	0.8372	**1.0**
SynB	**0.8457**	0.8437	0.7134	0.7862
SynC	**0.6498**	0.6302	0.3267	0.4079
SynD	0.8717	0.8247	0.7978	0.8112

Fig. 11. Adjusted Rand Index results w.r.t. base-line measures

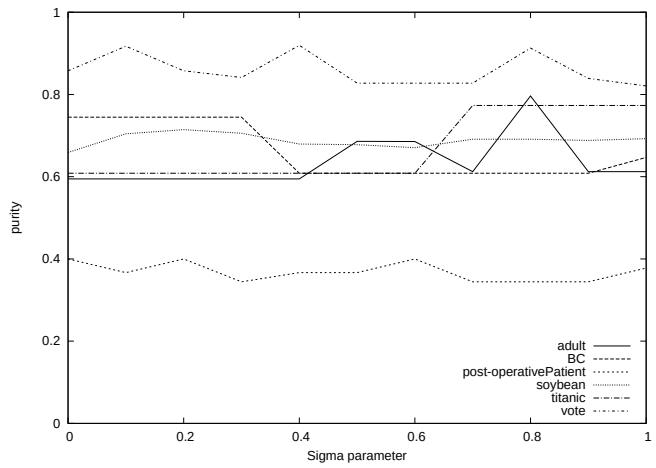
LIMBO and *ROCK* we set the parameters to zero. Using this setting we analyze the worst case for each of the methods. *DILCA*$_{RR}$ and *DELTA* do not need any parameter setting.

The first dimension under study is the number of instances. For this purpose, we generated 10,000 synthetic instances described by 50 attributes, then we built 10 datasets containing from 1000 to 10,000 instances. Results are shown in Figure 14(a). The picture shows that *DILCA* is faster than the other methods. *ROCK* was instead unable to process more than 8,000 instances within reasonable time.

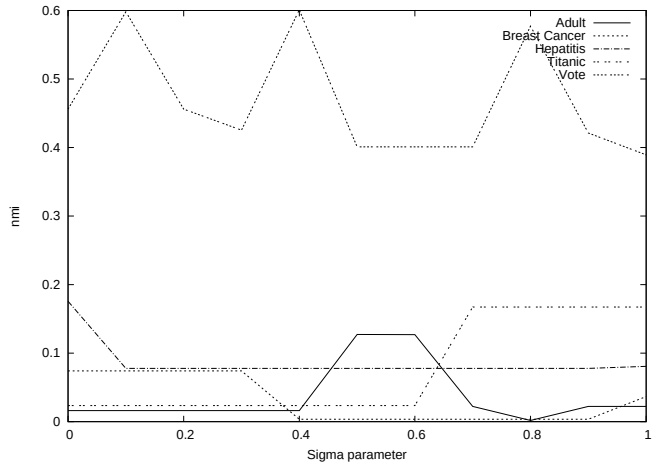
The second dimension under study is the number of features. We also performed an analysis on the scalability of the methods w.r.t. the number of features. We used another synthetic dataset consisting of 1,000 instances and 1,000 attributes, from which we built 10 datasets containing from 100 to 1,000 features. Each attribute ranges over ten nominal values. We compared the two versions of *DILCA* with *LIMBO* and *DELTA*. Unfortunately, it did not make much sense to analyze the scalability w.r.t. the number of features for $ROCK$, since it does not take into consideration a feature selection phase. Rather, the implementation provided by the authors takes as input already a point-to-point distance matrix, pre-computed on the complete feature set.

We report the results in Figure14(b). The picture shows again that *DILCA* is faster than the other methods. Let us now analyze the performance of *DILCA*$_M$ and *DILCA*$_{RR}$ in depth. Apparently they do not reflect the complexity analysis given in Section 4.3. However, as we anticipated, the complexity analysis supposes the worst case. This means that the time spent to rank all attributes is compensated by the second step of the algorithm. When the feature space grows, it is more probable that non-relevant and redundant features are present in the dataset. Indeed, during the second step, the algorithm only needs to compute distances over a small portion of the attribute set. To give an idea of the impact of the feature selection step, we report in Figure 15 the average number of retained features for the datasets described in Section 5.3.

Finally, we also investigated on the time spent by *DILCA*$_M$, *DILCA*$_{RR}$ and *DELTA* to perform clustering. In Figure 16 we report the time spent by the three parts of the hierarchical clustering algorithms: (i) computation of the distances between the values



(a)



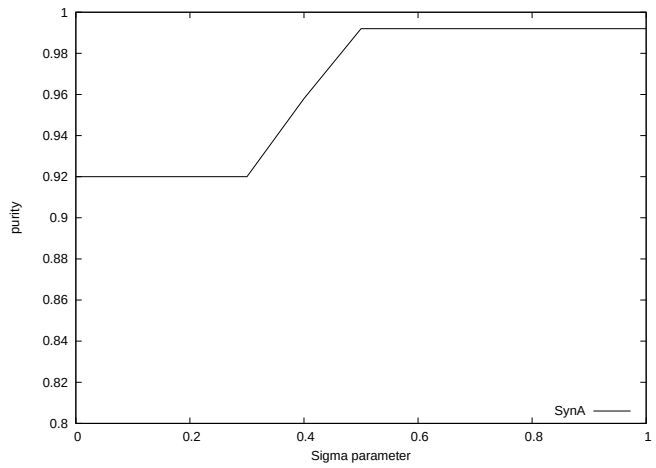
(b)

Fig. 12. Stability analysis on sigma parameter (UCI datasets)

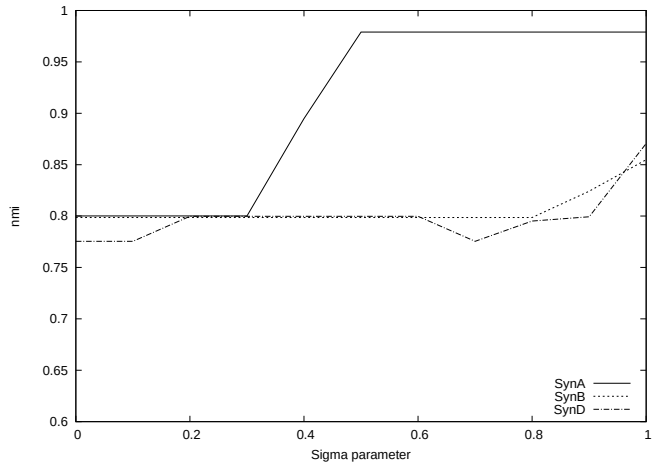
of the categorical attributes; (ii) computation of the point-to-point distance matrix; (iii) generation of the dendrogram.

Since *DELTA* is coupled with the same hierarchical clustering algorithm employed for *DILCA*, the second and third phases are the same for the three methods. Thus, we obtain a total of five curves that represent respectively the time spent by $DILCA_M$, $DILCA_{RR}$ and *DELTA* to compute distances between categorical values, the time spent to compute the instance-to-instance distance matrix given as input to the hierarchical algorithm, and the effective time spent by Ward algorithm to build the dendrogram.

As a result, we can notice that the most consistent portion of the overall computational time is employed to calculate the dendrogram and the instance-to-instance distance matrix. Notice also that *DELTA* performs much slower than both $DILCA_M$ and $DILCA_{RR}$.



(a)

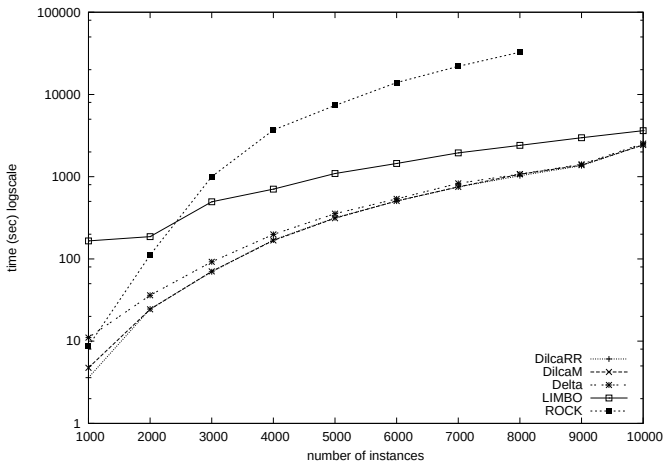


(b)

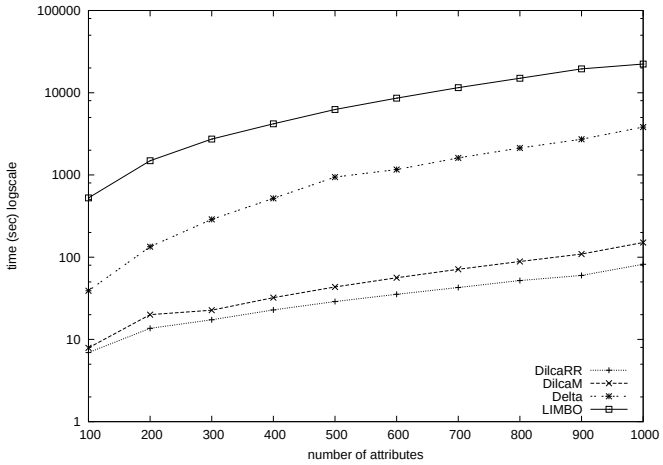
Fig. 13. Stability analysis on sigma parameter (Synthetic datasets)

6. CONCLUSION

We introduced a scalable approach to learn a context-based distance between the values of categorical attributes. We showed the effective impact of this approach on two distance-based clustering approaches. We believe that the proposed framework is general enough and it can be applied to any data mining task that involves categorical data and requires distance computations. As a future work we will investigate the application of our distance learning approach to different distance-based tasks. Although $DILCA_M$ seems preferable from the point of view of the computational complexity we must acknowledge that the computational complexity is given by the worst case. In practice, on average, experiments on real computational time showed that $DILCA_{RR}$ is preferable. In fact, the preliminary phase of context selection speeds up the distance computation phase. From the point of



(a)



(b)

Fig. 14. Time performance w.r.t. the number of instances (a) and attributes (b).

view of the quality of the obtained clusters whether $DILCA_M$ or $DILCA_{RR}$ is preferable is not easy to predict. It is a matter of future work to learn which of them would be better as a prediction task according to the typology of the data.

A first possible application of $DILCA$ is within a nearest neighbors classifier (kNN [Bishop 2006]) to estimate a good distance measure between two instances that contain categorical attributes. Moreover, any other distance-based classification algorithm is a good candidate for $DILCA$, e.g., SVM [Bishop 2006].

Boolean variables are a special case of categorical attributes. $DILCA$ can be used with any task that employs boolean features and the necessity of computing distance between instances, e.g., within a transactional clustering [Yang et al. 2002] algorithm. In transactional clustering we can represent a transaction as a boolean vector that indicates if a spe-

Dataset	Orig. # of attr.	Avg. # of attr. in context
Audiology	59	5.55 ± 5.25
Vote	16	2.94 ± 1.34
Mushroom	22	4.36 ± 1.97
Dermatology	34	4.24 ± 1.78
Soybean	35	4.71 ± 1.68
Car	6	3.0 ± 0.0
Adult	14	2.93 ± 1.10
Post-operative	8	2.25 ± 0.97
Titanic	3	1.33 ± 0.47
Breast-Cancer	9	2.33 ± 0.94
Ballons	4	2 ± 1
SynA	50	6.92 ± 2.58
SynB	50	6.86 ± 2.20
SynC	100	10.54 ± 2.75
SynD	100	10.13 ± 3.18

Fig. 15. Mean number of attributes in the context for *DILCA*$_{RR}$

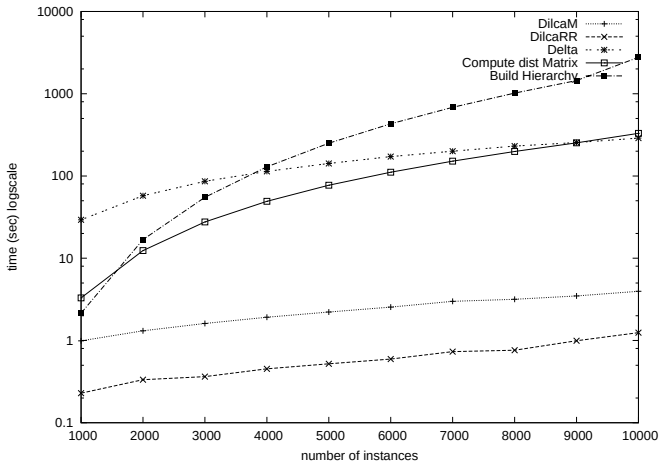


Fig. 16. Computational time for each hierarchical clustering phase.

cific object is contained in the transaction. Another field of application is anomaly/outlier detection [Tan et al. 2005].

Finally, using this distance it will be possible to compute distances between objects described by both numerical and categorical attributes.

ACKNOWLEDGMENTS

The authors wish to thank Dr. Eui-Hong Han who provided the source code $ROCK$, Dr. Periklis Andritsos who provided the implementation of $LIMBO$, and Elena Roglia for stimulating discussions. We want to thank Regione Piemonte which co-funds Ruggero G.

Pensa.

REFERENCES

AHMAD, A. AND DEY, L. 2007. A method to compute distance between two categorical values of same attribute in unsupervised learning for categorical data set. *Pattern Recogn. Lett. 28,* 1, 110–118.

ANDRITSOS, P., TSAPARAS, P., MILLER, R. J., AND SEVCIK, K. C. 2004. Scalable clustering of categorical data. In *Proc. of EDBT 2004*. Springer, Heraklion, Crete, Greece, 123–146.

BARBARA, D., COUTO, J., AND LI, Y. 2002. Coolcat: an entropy-based algorithm for categorical clustering. In *Proc. of CIKM 2002*. ACM Press, McLean, VA, USA, 582–589.

BISHOP, C. M. 2006. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

BLAKE, C. L. AND MERZ, C. J. 1998. UCI repository of machine learning databases. http://www.ics.uci.edu/ mlearn/MLRepository.html.

BORIAH, S., CHANDOLA, V., AND KUMAR, V. 2008. Similarity measures for categorical data: A comparative evaluation. In *SDM*. 243–254.

GANTI, V., GEHRKE, J., AND RAMAKRISHNAN, R. 1999. Cactus-clustering categorical data using summaries. In *Proc. of ACM SIGKDD 1999*. ACM Press, San Diego, CA, USA, 73–83.

GUHA, S., RASTOGI, R., AND SHIM, K. 1999. Rock: A robust clustering algorithm for categorical attributes. In *Proc. of IEEE ICDE 1999*. IEEE Computer Society, Sydney, Austrialia, 512–521.

GUYON, I. AND ELISSEEFF, A. 2003. An introduction to variable and feature selection. *J. Mach. Learn. Res. 3*, 1157–1182.

HAN, J. AND KAMBER, M. 2000. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann.

HUANG, Z. 1998. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov. 2,* 3, 283–304.

HUBERT, L. AND ARABIE, P. 1985. Comparing partitions. *Journal of Classification 2,* 1, 193–218.

IENCO, D., PENSA, R. G., AND MEO, R. 2009. Context-based distance learning for categorical data clustering. In *Proceedings of the 8th International Symposium on Intelligent Data Analysis, IDA 2009*. Lecture Notes in Computer Science, vol. 5772. Springer, Lyon, France, 83–94.

KASIF, S., SALZBERG, S., WALTZ, D. L., RACHLIN, J., AND AHA, D. W. 1998. A probabilistic framework for memory-based reasoning. *Artif. Intell. 104,* 1-2, 287–311.

LI, T., MA, S., AND OGIHARA, M. 2004. Entropy-based criterion in categorical clustering. In *Proc. of ICML 2004*. Banff, Alberta, Canada, 536–543.

MELLI, G. 2008. Dataset generator, perfect data for an imperfect world. http://www.datasetgenerator.com.

QUINLAN, R. J. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann.

STANFILL, C. AND WALTZ, D. L. 1986. Toward memory-based reasoning. *Commun. ACM 29,* 12, 1213–1228.

STREHL, A., GHOSH, J., AND CARDIE, C. 2002. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research 3*, 583–617.

TAN, P.-N., STEINBACH, M., AND KUMAR, V. 2005. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

VERDUGO, L. A. AND RATHIE, P. N. 1978. On the entropy of continuous probability distributions. *IEEE Transactions on Information Theory 24*, 120–122.

WITTEN, I. H. AND FRANK, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*, 2 ed. Data Management Systems. Morgan Kaufmann.

YANG, Y., GUAN, X., AND YOU, J. 2002. Clope: A fast and effective clustering algorithm for transactional data. In *Proc. of ACM SIGKDD 2002*. ACM Press, Edmonton, Alberta, Canada, 682–687.

YU, L. AND LIU, H. 2003. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of ICML 2003*. Washington, DC, USA, 856–863.

ZAKI, M. J. AND PETERS, M. 2005. Clicks: Mining subspace clusters in categorical data via k-partite maximal cliques. In *Proc. of IEEE ICDE 2005*. IEEE, Tokyo, Japan, 355–356.