From Categorical to Numerical: Multiple Transitive Distance Learning and Embedding

Kai Zhang*† Qiaojun Wang‡ Zhengzhang Chen§¶ Ivan Marsic‡ Vipin Kumar‖
Guofei Jiang§ Jie Zhang**

Abstract

Categorical data are ubiquitous in real-world databases. However, due to the lack of an intrinsic proximity measure, many powerful algorithms for numerical data analysis may not work well on their categorical counterparts, making it a bottleneck in practical applications.

In this paper, we propose a novel method to transform categorical data to numerical representations, so that abundant numerical learning methods can be exploited in categorical data mining. Our key idea is to learn a pairwise dissimilarity among categorical symbols, henceforth a continuous embedding, which can then be used for subsequent numerical treatment. There are two important criteria for learning the dissimilarities. First, it should capture the important "transitivity" which has shown to be particularly useful in measuring the proximity relation in categorical data. Second, the pairwise sample geometry arising from the learned symbol distances should be maximally consistent with prior knowledge (e.g., class labels) to obtain a good generalization performance. We achieve them through multiple transitive distance learning and embedding. Encouraging results are observed on a number of benchmark classification tasks against state-of-the-art.

1 Introduction

Categorical data are often encountered in practical learning problems [1, 2, 8]. Unlike numerical variables that can take values arbitrarily in the real domain, a categorical variable (or sometimes called nominal variable) can only take one of a limited number of possible values or levels, such as the blood type of a person (A, B, AB or O), the weather condition (windy, cloudy or rainy) [3, 12], and severity of a symptom (mild, moderate, or severe). These possible values will be referred to as "symbols" throughout this paper. Categorical symbols usually do not have any intrinsic ordering, and they can not be simply treated using algebraic operations, thus many popular numerical learning algorithms are not directly applicable. On the other hand, even the categorical symbols can be turned into numbers with encoding schemes, algorithms designed for numerical data may give poor results on the categorical counterparts. Both are bottlenecks for categorical data analysis.

We believe these difficulties arise from the fact that there lacks a well-defined distance between categorical symbols. As a result, the distances between samples are hard to compute, so is the geometry of sample distributions. While in many popular learning algorithms, the pairwise sample distance/similarity is the key to the learning results, such as support vector machines [6, 24], spectral clustering [13], Gaussian processes [14], and manifold learning [5, 10, 19, 23].

Motivated by this observation, in this paper we pursue the interesting topic of transforming categorical data into numerical ones to resolve the aforementioned difficulties in categorical data analysis. The key objective is to find a numerical representation of the categorical data, i.e., each symbol in the data is replaced with a number (or a r-dimensional numerical vector), and as a result many numerical learning and data mining algorithms can be readily applied on them. In order to achieve this, we need to first learn a pairwise dissimilarity measure on the categorical symbols. Once the distance between symbols is defined, it can be used to compute the distances between samples[1], and many

*School of Computer Science, Southwest Petroleum University, Chengdu, Sichuan Province, P.R. China 610500
†Institute of Data Science and Technology, Alibaba Group, Seattle, WA 98101
‡Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, NJ 08854
§NEC Laboratories, America Inc., Princeton 08540, NJ
¶Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208
‖Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455
**Center for Computational Systems Biology, Fudan University, Shanghai, P.R. China 200433

[1] As will be clear in Section 2, the distance between two samples is simply the sum of the distances between their respective attributes across all dimensions, which is similar to the custom in handling numerical (Euclidean) data.

learning algorithms can therefore be readily applied.

Another important advantage of considering the pairwise symbol dissimilarity is that, equivalently, it leads to a continuous embedding using techniques of manifold learning [5, 10, 19, 23]. In other words, each symbol is then endowed with a numerical representation, which allows them to be treated exactly as numerical data. This can significantly broaden the applicability of many learning algorithm in categorical data analysis, and can also bring more computational efficiency than explicitly computing pairwise sample distances.

Embedding of the categorical symbols also has the advantage of giving a data-driven visualization for better understanding the relation between symbols. Note that categorical symbols are defined by human experts, which can be subjective and in many cases lack a strict, quantitative base. It can be very hard to tell the similarity or difference between two symbols. While the embedding of the categorical symbols in our approach is based on the co-occurrence behavior of all the symbols as well as the class labels, which will faithfully reflect the relationship between the categorical symbols with particular respect to the learning task at hand (such as classification). We believe it is very valuable for domain experts to combine the data-driven embedding with the domain knowledge. For example, by examining the embedding results as shown in Figure 1, human experts will have a direct intuition on how the symbols are similar to each other, so as to explore their relations or even refine their definitions.

The pairwise symbol dissimilarities should capture intrinsical structures in the data, in order for the resultant embedding to be a useful one. To achieve this, we propose two important learning criteria. First, the proximity relation underlying the dissimilarities should be "transitive". Specifically, the relation between two symbols should be determined by globally taking into account their relation with other symbols, so that the "closeness" can be systematically transmitted among symbols. Such transitivity has been shown that particularly useful in measuring the proximity relation among categorical variables [18]. Second, the pairwise sample geometry arising from such symbol distances should be maximally coincident with prior knowledge, such as the class labels, to obtain a good generalization performance. In particular, it is preferred that embedded samples will be compact within one class, while well separated among different classes.

To satisfy these two criteria, we propose a novel, multiple distance learning and embedding approach to obtain a mixture of transitive symbol distances using the class labels as a guidance. The resultant mixture distance captures both unsupervised and supervised information about the data. The unsupervised information is related to co-occurrence statistics of the categorical symbols, which is summarized into a number of bipartite graphs based on different choices of the proximity measure; the supervised information is related to the class labels, which serves as a guidance on how the different proximities (or graphs) should be combined together to maximize the generalization performance. To the best of our knowledge, this is the first work attempts to find numerical representations of categorical variables through (semi-)supervised learning framework. Our method opens up the possibility to exploiting the abundant, numerical learning algorithms for categorical data analysis. Encouraging results are observed on a number of benchmark classification tasks against state-of-the-art.

The rest of the paper is organized as follows. Section 2 introduces some necessary definitions and formally defines the problem. In Section 3, we discuss the embedding of categorical variables using various types of transitive distance measures. In Section 4, we propose a multiple distance learning framework to combine the different base-distances calculated in Section 3. Section 5 discusses several related work in handling categorical symbols in clustering tasks. Section 6 reports the empirical evaluations, and compares the proposed method with both baseline coding schemes and state-of-the-art methods in transforming categorical data into numerical ones. In Section 7, we make concluding remarks and point out some interesting future directions.

2 Problem Statement

Suppose we have a categorical data represented as a $n \times d$ symbolic matrix $\overline{X}$, where n is the sample size and d is the number of features (or attributes). We also have the class labels $\mathbf{y} \in \{1, -1\}^n$. Let A_j be the symbols used in the jth feature, where $|A_j| = c_j$, and A be the set of all symbols, i.e., $A = \cup_{j=1}^{d} A_j$, and $|A| = \sum_{j=1}^{d} |A_j| = c$. Our goal is to learn a pairwise distance measure $S \in \mathbb{R}^{c \times c}$ on all the symbols, or a r-dimensional embedding, $\mathbf{e} \in \mathbb{R}^{c \times r}$, such that a good generalization performance can be obtained using the learned distance/embedding for training and testing.

To simplify notations and express our main idea more conveniently, let A be an ordered set, i.e.,

$$A = \{\underbrace{a_{l_1+1}, a_{l_1+2}, ...}_{A_1}, \underbrace{a_{l_2+1}, a_{l_2+2}, ...}_{A_2}, ..., \underbrace{a_{l_d+1}, a_{l_d+2}...}_{A_d}\}, \tag{2.1}$$

where $l_j = \sum_{k=1}^{j-1} c_k$, for $j \geq 2$, and $l_1 = 0$. Also, we transform the categorical data $\overline{X}$ into an integer matrix X as follows. Let $A(k)$ denote the kth symbol in A (Eq.

2.1). Then

$$X_{ij} = k \text{ if } \overline{X}_{ij} = A(k). \tag{2.2}$$

Here, $\overline{X}_{ij}$ denotes the jth symbolic feature of the ith instance. Namely every symbol in $\overline{X}$ is represented by its position in A. The reason to use such an integer representation is that it can greatly simplify the indexing in our derivations.

In manipulating the distance between symbolic variables, we follow the convention of the L_θ-norm distance. That is, the L_θ distance between two instances is the sum of the L_θ distance along each dimension. In particular, the distance between the ith and jth instance in X can be represented as

$$\text{dis}\,(X_i, X_j)^\theta = \sum_{k=1}^{d} \text{dis}\,(X_{ik}, X_{jk})^\theta . \tag{2.3}$$

This assumption makes our derivations tractable, which also seems natural for categorical symbols that will be endowed with numerical representations. In practice, we can choose θ as 1 or 2, which bears the resemblance to Hamming distance and Euclidean distance, respectively.

3 Symbolic Embedding Via Transitive Proximities

Categorical symbols do not have any intrinsic ordering associate with them. Therefore, in order to obtain a numerical representation for categorical symbols, we will first resort to computing the pairwise proximity among the symbols, and then compute a Euclidean embedding that recovers such a proximity.

3.1 Transitive Proximity Co-occurrence is probably the most direct way to measure the relation between categorical symbols. However, the co-occurrence statistics has one limitation that it is not transitive. For example, symbol a often co-occurs with b, and b often co-occurs with c, but a and c seldom co-occur. Then, based on the co-occurrence statistics, a and c are not close (or very dissimilar). However, arguably, both a and c are indirectly connected by b, therefore they should also share certain level of similarity.

To better understand this, consider using a graph to encode the co-occurrence structures among the symbols in A, where each node represents a symbol and an edge denotes co-occurrence. Since symbols belong to the same dimension would never appear together, we will naturally have a d-partite graph, each partition containing the alphabets of one dimension of the data, A_i, for $i = 1, 2, ..., d$. Based on this graph, symbols in the same A_i's are considered having a similarity 0 since they will not be connected with each other by any edge. Now, imagine we compute the distance between two instances (as in Eq. 2.1), then we would need the distance between symbols in the same dimension A_i, while it appears difficult to define the distance between two symbols with zero similarity. In other words, the co-occurrence itself is an "incomplete" proximity.

In order to "augment" the co-occurrence based similarity, we propose to use the shortest path distance on the d-partite graph as a new proximity measure. Note that each node of the d-partite graph represents a symbol in A. Therefore, the shortest path distance between every pair of nodes, $A(p)$ and $A(q)$, provides a systematic way to augment the initial symbol relation by collectively considering their relation with other symbols. In particular, symbols belonging to the same dimension A_i can now be related with each other by their connections with other dimensions. Therefore the proximity relation is transitive. We use a matrix $S^\theta \in R^{c\times c}$ to denote the shortest path distance, whose (p, q)th entry is

$$S^\theta_{(p,q)} = \text{dis}(A(p), A(q))^\theta, \quad 1 \le p, q \le c. \tag{3.4}$$

where θ is a power parameter. We call S θ-norm symbolic distance matrix.

Construction of the d-partite Graph. The initial d-partite graph is constructed as follows. Suppose we have c nodes, each represents one symbol in A. Then we link all pairs of nodes $A(p)$ and $A(q)$ for $1 \le p, q \le c$ that belong to different dimensions. By doing this, the resultant graph is always d-partite. In table 1, we list different choices for computing the edge weights. Here, $\mathbf{a}_i \in \{0, 1\}^{n\times 1}$ is a vector of 1/0's recording the occurrence of the ith symbol with all the n instances, for $i = 1, 2, ..., c$.

Note that the proximity listed in Table 1 can be either in terms of "similarity" or "distance". In order to compute the shortest path distance, however, we will need to transform the similarity to distance. For zero similarity, the corresponding distance will be infinity. For non-zero similarity, we can use different transfer functions: $(1)f(x) = 1$; $(2)f(x) = \frac{1}{x}$; and $(3)f(x) = -\log(x)$. In some cases, one might want to further sparsify the d-partite graph by removing those connections that are not within k-nearest neighbors. Here, k is an integer bounded by d.

One might want to apply the distances defined in Table 1 (Euclidean or Hamming) to directly compute a dense, pairwise distance matrix, in which even symbols with zero-occurrence can still have a finite distance among them. However, this can be less robust, because distances between symbols that are not very correlated

Table 1: Edge weights for different proximity measures.

	Proximity	Connected Edge	Non-connected Edge
Similarity	Co-occurrence	$\lvert\mathbf{a}_p \cap \mathbf{a}_q\rvert$	
	Normalized occurrence	$\lvert\mathbf{a}_p \cap \mathbf{a}_q\rvert / \lvert\mathbf{a}_p \cup \mathbf{a}_q\rvert$	
	Mutual information	$\sum_{e,\tilde{e}\in\{0,1\}} p(\mathbf{a}_p = e, \mathbf{a}_q = \tilde{e})$ $\times \log\left(\frac{p(\mathbf{a}_p=e,\mathbf{a}_q=\tilde{e})}{p(\mathbf{a}_p=e)\cdot p(\mathbf{a}_q=\tilde{e})}\right)$	0
Distance	Hamming distance	$\lvert\mathbf{a}_p - \mathbf{a}_q\rvert$	
	Euclidean distance	$\lVert\mathbf{a}_p - \mathbf{a}_q\rVert_2$	+infinity
	Cosine distance	$\arccos(\mathbf{a}_p'\mathbf{a}_q / \sqrt{\lVert\mathbf{a}_p\rVert \cdot \lVert\mathbf{a}_q\rVert})$	

may no longer faithfully reflect their proximities, and in practice, not all symbols are closely related to each other. In comparison, by only connecting very correlated symbols and then using shortest path to link less-correlated symbols, we expect to obtain much more robust proximity.

3.2 Proximity Preserving Embedding After computing the pairwise symbol distance matrix S^θ (as in Eq. 3.4), we can then find a r-dimensional embedding coordinates $\mathbf{e}_i$'s that preserve the distances, i.e.,

$$\|\mathbf{e}_p - \mathbf{e}_q\|_2^2 \approx S^\theta_{(p,q)}. \tag{3.5}$$

This is often called manifold learning (or dimension reduction). Various approaches have been proposed to compute a non-linear representation that captures the data proximity relations. Most of these approaches first compute a pairwise similarity (or distance) between objects, and then obtain the embedding results through an eigenvalue decomposition.

For example, if S^θ is chosen as squared Euclidean distance, then one can find embedding (as in Eq. 3.7) that exactly recovers such distances, via so called multi-dimensional scaling [5]. To obtain such an embedding, one computes the eigenvalue decomposition of the following matrix

$$-\frac{1}{2}HSH = U\Lambda U, \tag{3.6}$$

where H is the double centering matrix, U has columns as the eigenvectors and Λ is a diagonal matrix with eigenvalues. Then the embedding $E = [\mathbf{e}_1 \ \ \mathbf{e}_2 \ \ ...; \mathbf{e}_c]'$ can be chosen as

$$E = U_k \Lambda_k^{\frac{1}{2}}. \tag{3.7}$$

In [10], the pairwise distance between objects is chosen as the geodesic distance (approximated by the shortest path distance on a pre-computed graph), leading to the well known ISOMAP. In this case, since the distances in S are obtained in non-Euclidean manner, matrix in (Eq. 3.6) can be indefinite and the embedding only approximately recovers the distances in S.

Note that we also used the shortest path distances as discussed in Subsection 3.1. However, we believe that using the shortest path distance to obtain a "transitive" distance measure between categorical symbols is a novel application. In addition, instead of using only one distance measure, we compute the mixture of a number of "base" shortest-path distances with the guidance of the label information, which corresponds to a more generalized, semi-supervised setting.

4 Learning with Multiple Transitive Distances

As discussed in Section 3.1, in constructing the d-partite graph, one can choose different proximity relations or the neighborhood size k. It is unclear which choice leads to the best proximity measure, and how to perform model selection is typically a challenging task. In case of unsupervised learning, since there is no label information, one may have to make choices empirically. When class labels are available, we can use them as a guidance for model selection. Here, inspired by the works of multiple kernel learning, we propose a multiple distance learning scheme to solve the model selection problem in our setting.

4.1 Multiple Kernel Learning In the literature on the SVM and kernel methods [23], researchers have emphasized the need to consider multiple kernels instead of a single kernel matrix. This can improve the model flexibility, and also reflect the practical situation that learning problems often involve multiple, heterogeneous data sources (aspects). Usually, the learning procedure will compute a conic combination of a set of base ker-

nels K_i's, where the mixing coefficients are obtained by simultaneously optimizing a performance measure corresponding to the generalization performance of classifiers [20, 21, 22, 25]. For example, [20] used the following SDP formulation

$$\begin{aligned} \min \quad & \omega(K) \\ & \text{s.t. } \operatorname{trace}(K) = c, \\ & K \succeq \mathbf{0}, \;\; K = \textstyle\sum_{i=1}^{m} \mu_i K_i, \end{aligned}$$

where $\omega(K)$ is the performance measure corresponding to an SVM with dual variables v, defined as $\omega(K) = \max_{\mathbf{v}} 2\mathbf{v}^\top \mathbf{e} - \mathbf{v}^\top \left(\mathbf{y}\mathbf{y}^\top \bigodot K + \tau \cdot I\right) \mathbf{v}, 0 \leq \mathbf{v} \leq C, \mathbf{v}^\top \mathbf{y} = 0$. Here, C is the regularization (const) parameter, I is the identity matrix. The resultant problem can be solved by a second-order cone programming. Later more efficient formulation called SimpleMKL [22] was proposed to obtain sparse kernel combinations.

4.2 Multiple Transitive Distance Learning Inspired by the multiple kernel learning framework, we apply similar idea in dealing with multiple distance matrices as computed in Section 3. Suppose we have obtained a number of symbol distance matrices $S_k^\theta \in \mathbb{R}^{c\times c}$ for $k = 1, 2, ..., L$, each of which is computed using different choices of proximity or neighborhood size (or a combination). Note that these "base" distance matrices may reflect different aspect of the data structures. For example, a larger neighborhood size reveals long term interactions among symbols, making the resultant pairwise distance matrix denser, while a smaller neighborhood size relates to short range interactions, leading to sparser distance matrices. On the other hand, the choice of the different proximity measure (Table 1) will also have an impact on the resultant base distance matrices.

Then, our goal is to obtain a weighted average of all these "base" distances, in the form of

$$S^\theta = \sum_{m=1}^{L} \alpha_m S_m^\theta, \tag{4.8}$$

where α_m's are non-negative coefficients. To make the mixture distance useful in practical classification tasks, a natural criterion is that the resultant embedding using the mixture distance S^θ should lead to desirable learning performance.

To see how S^θ (or S_m^θ's) will practically affect the learning performance of a future classifier, we need to transform them to pairwise sample distances, the latter fully determining the geometry of samples hence the output of a classifier. Let the pairwise sample distances be $D^\theta \in \mathbb{R}^{n\times n}$. Then its (i, j)th entry, i.e. the power-θ distance between ith and jth sample can be written as

$$\begin{aligned} D_{(i,j)}^\theta &= \operatorname{dis}(X_i, X_j)^\theta \\ &= \sum_{k=1}^{d} \operatorname{dis}(X_{ik}, X_{ik})^\theta \\ &= \sum_{k=1}^{d} S_{(X_{ik},X_{jk})}^\theta \\ &= \sum_{k=1}^{d} \sum_{m=1}^{L} \alpha_m S_{m(X_{ik},X_{jk})}^\theta \\ &= \sum_{m=1}^{L} \alpha_m D_{m(i,j)}^\theta \end{aligned}$$

where we have used the relation in Eq. 2.3, and

$$D_{m(i,j)}^\theta = \sum_{k=1}^{d} S_{m(X_{ik},X_{jk})}^\theta. \tag{4.9}$$

In other words, the pairwise sample distance matrix D^θ can also be deemed as the mixture of a set of base sample distance matrices, D_m^θ's, each arising from a base symbol distance matrix S_m^θ for $m = 1, 2, ..., L$. Consequently, we have a mixed sample distance matrix in the form of

$$D^\theta = \sum_{m=1}^{L} \alpha_m D_m^\theta. \tag{4.10}$$

The advantage of using expression (Eq. 4.10) is that, since D_m^θ's can be readily computed using S_m^θ's, and D^θ is directly associated with the learning performance, our problem can be easily formulated as an optimization of the mixing coefficients α_m's to obtain desired distributions of embedded samples.

Instead of using the learning criterion in multiple kernel learning [20, 21, 22] that is associated with the performance of a specific classifier (SVM), here we want to adopt a general criterion to optimize the learning with multiple distances. In the machine learning community, one such criterion is that samples belonging to the same class should be close to each other; while samples from different classes should be far away from each other[16]. Based on this idea, in the following we propose a discriminative embedding approach to optimize mixing coefficients α_m's, such that the resultant sample distance matrix D^θ will have small intra-class distances and large inter-class distances. Note that the mixing coefficients have been shown to apply to both the sample distance matrices D_m^θ's (Eq. 4.10) and the symbol distance matrices S_m^θ's (Eq. 3.4). Therefore, after learning α_m's, we can finally obtain the mixed distance S^θ (Eq. 3.4), based on which a numerical embedding for all symbols can be computed as in Section 3.2.

4.3 Discriminative Embedding Suppose we are given a set of must link and cannot link constraints, $\mathcal{S}$ and $\mathcal{D}$, respectively. To measure the intra-class compactness, define $J_{\mathcal{S}}$ as the the sum of all distances from the same-class-pairs (subject to power ϵ). Define $\alpha = [\alpha_1 \quad \alpha_2 \quad ... \quad \alpha_L]^\top$, and $\mu_{ij}^\theta = [D_{1(i,j)}^\theta \quad D_{2(i,j)}^\theta \quad ... \quad D_{L(i,j)}^\theta]^\top$. Then $J_{\mathcal{S}}$ can be written as

$$\begin{aligned} J_{\mathcal{S}}^{(\theta,\epsilon)} &= \sum_{(i,j)\in\mathcal{S}} \left(D_{(i,j)}^\theta\right)^\epsilon \\ &= \sum_{(i,j)\in\mathcal{S}} \left(\sum_{m=1}^{L} \alpha_m D_{m(i,j)}^\theta\right)^\epsilon \\ &= \sum_{(i,j)\in\mathcal{S}} \left((\mu_{ij}^\theta)^\top \alpha\right)^\epsilon. \end{aligned}$$

Similarly, define $J_{\mathcal{D}}^{(\theta,\epsilon)}$ as the the sum of distances from the different-class-pairs (with power ϵ).

$$J_{\mathcal{D}}^{(\theta,\epsilon)} = \textstyle\sum_{(i,j)\in\mathcal{D}} \left((\mu_{ij}^\theta)^\top \alpha\right)^\epsilon.$$

If we want to maximize $J_{\mathcal{D}}^{(\theta,1)} - J_{\mathcal{S}}^{(\theta,1)}$, we have the following linear programming problem:

$$\begin{aligned} (4.11)\quad \max_{\alpha} \quad & \left(\textstyle\sum_{(i,j)\in\mathcal{D}} \mu_{ij}^\theta - \sum_{(i,j)\in\mathcal{S}} \mu_{ij}^\theta\right)^\top \alpha \\ s.t. \quad & \alpha \geq 0, \alpha^\top \mathbf{1} = 1 \end{aligned}$$

One can also choose to minimize the ratio $\frac{J_{\mathcal{S}}^{(\theta,2)}}{J_{\mathcal{D}}^{(\theta,2)}}$. Then we have the problem:

$$\min \frac{\alpha^\top \left(\sum_{(i,j)\in\mathcal{S}} \mu_{ij}^\theta (\mu_{ij}^\theta)^\top\right) \alpha}{\alpha^\top \left(\sum_{(i,j)\in\mathcal{D}} u_{ij}^\theta (\mu_{ij}^\theta)^\top\right) \alpha}.$$

Define the following variables,

$$\begin{aligned} A_\theta &= \sum_{(i,j)\in\mathcal{S}} \mu_{ij}^\theta (\mu_{ij}^\theta)^\top \\ B_\theta &= \sum_{(i,j)\in\mathcal{D}} \mu_{ij}^\theta (\mu_{ij}^\theta)^\top \end{aligned}$$

let $B_\theta^{\frac{1}{2}} \alpha = \beta$, then we have the following problem,

$$\begin{aligned} (4.12)\quad \min \quad & \beta^\top \left(B_\theta^{-\frac{1}{2}} A_\theta B_\theta^{-\frac{1}{2}}\right) \beta \\ \text{s.t.} \quad & B_\theta^{-\frac{1}{2}} \beta \geq 0 \\ & \beta^\top B_\theta^{-\frac{1}{2}} \mathbf{1} = 1. \end{aligned}$$

This is a standard quadratic programming (QP), for which a global optimal solution can be obtained efficiently in polynomial time. In practice one can choose $\theta = 1$ or $\theta = 2$, which corresponds to the original and squared shortest-path-distance in Eq. 3.4.

5 Related Work

Categorical data are often observed in data mining tasks, and researchers have proposed to use coding systems to turn categorical variables into numbers for analysis [11]. The coding scheme should minimize redundancy while still representing the complete data set. The most popular choice is the dummy variable coding (DVC). Symbols in the ith feature, A_i, are coded with c_i-dimensional vectors ($|A_i| = c_i$), where each vector has a single 1 corresponding to that symbol, and all rest entries are 0's. By doing this, the original data $\overline{X}$ will be transformed to c-dimensional numerical data, where $c = \sum_{i=1}^{d} |A_i|$. Although such a coding scheme is easy to implement, it assumes that the distance among all the symbols equals to 1. Considering that the symbols used to represent the data can have various meanings and levels, this assumption obviously deviates from the truth.

In [17], the authors proposed a density-based logistic regression (DLR) framework. The DLR maps the data to a feature space using $p(y|x_d)$, the posterior probability of the attribute x_d belonging to class with label y. In case of continuous data, a kernel density estimator needs to be computed whose bandwidth parameter is jointly optimized with the logistic regression procedure. In case of categorical data, this then becomes replacing each symbol with the histogram of class labels associated with that symbol. Such a transformation also imposes a distance measure between categorical symbols, namely, if two symbols are associated with the same distribution of class labels, then they will have the same representation and henceforth a zero distance between them.

In [18], the authors proposed a dynamical system approach to analyze relation between categorical symbols and to group them into clusters. The basic idea is to iteratively apply a pre-defined operation $f(\cdot)$ on the weights associated with each symbol, until a stationary state of the system is reached. It can be shown that under certain conditions, the iteration simulates computing the eigenvectors of the similarity matrix among symbols. The idea of enforcing interactions among symbols through iterative updates is very interesting, and our approach achieves similar goal through the use of the shortest path distances. However, an important difference is that instead of grouping the symbols, we are aimed at learning an embedding of symbols that will also be aligned with the class labels.

There are also a large number of algorithms that are specifically devoted to clustering categorical data [7, 9]. For example, in [7], a novel concept of "links" is proposed to measure the similarity between two data points in the form of transaction data. The number

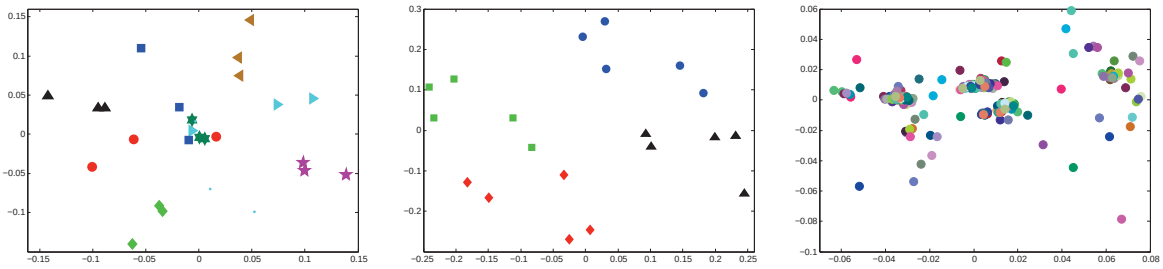

Figure 1: Embedding of the categorical symbols in two-dimensional space for data set Tic-tac-toe, Balance, and Splice, respectively. Each color represents symbols from the same attributes.

of links between two points is the number of common neighbors. Therefore, this criterion incorporates global information in computing the sample distances, which is more robust. However, it still requires an initial distance measure between data points to determine the neighbors of each data point, which again relies on some pre-defined distance between symbols and is an open problem itself. Note that these types of work focus on clustering while our goal is to find a numerical representation of the categorical data, which can be used for many different tasks besides clustering.

More algorithms on handling the categorical data can be found in [15]. Note that the difference of our approach and these algorithms is that instead of designing a specific algorithm to analyze the categorical data, we instead seek a transform from categorical to the numerical domain such that a wide variety of existing numerical algorithms can be better applied in categorical data analysis. A byproduct of our approach is the visualization of the symbolic attributes, which naturally captures the statistical properties of the data and can provide more insight on the sematic level. This can be quite useful in improving the quantitative analysis in areas of psychology, behaviourial analysis, and marketing analysis.

6 Experiments

In this section, we compare the following algorithms for categorical data classification: (1)Dummy variable coding; (2)Density-based coding [17]; (3)Decision tree algorithm [4]; and (4)Our approach. Here methods (1), (2) and (4) are coding schemes that transform categorical variables into numerical values, and method (3) is an algorithm that is specifically designed (and particularly suited) for categorical data mining and has been extremely popular in the literature. The classifier used to evaluate the quality of classification is logistic regression. The logistic regression and decision tree classifier are adapted from open source project Scikit-Learn[2]. The QP solver used in our approach is the cvx package. Our codes are written in Python and run on a Intel(R) Core(TM) i5 CPU @2.60GHZ 2.60GHZ PC with 8 GB RAM.

Table 2: The statistics of the benchmark data sets.

Data	# inst.	#dim.	#symbols
Balance	525	4	20
Mushroom	8,124	22	117
Tic-Tac-Toe	958	9	27
Splice	3,190	60	240
Cancer	296	9	89
Hayes-Roth	160	4	15
Monk	432	6	17

We use the following benchmark data sets from UCI Machine Learning Repository[3], which are briefly summarized as follows:

- Balance Scale: This data set was generated to model psychological experimental results. Each example is classified as having the balance scale tip

[2]http://scikit-learn.org/stable/
[3]https://archive.ics.uci.edu/ml/datasets.html

Table 3: The mean and standard deviation of the classification accuracies (in %) for different algorithms.

	Balance	Mushroom	Tic-Tac-Toe	Splice	Cancer	Hayes-Roth	MONK
Dummy	98.50	99.04	95.25	91.18	95.81	77.42	96.24
	0.75	0.73	0.42	0.94	1.23	6.82	1.13
Density	96.59	98.43	70.60	**91.29**	**96.76**	57.72	96.37
	1.58	0.57	2.34	**0.98**	**0.76**	8.94	0.85
Decision	88.49	**99.40**	87.48	88.68	94.25	73.93	97.13
Tree	1.63	**0.54**	2.18	1.73	1.38	7.74	0.72
Ours	**99.42**	**99.54**	**98.25**	**91.51**	95.94	**83.91**	**97.81**
	0.58	**0.46**	**0.44**	**1.20**	0.93	**3.64**	**1.23**

to the right, tip to the left, or be balanced. There are 525 instances, 4 attributes, and 20 symbols.

- Mushroom: This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms. There are 8,124 instances, 22 attributes, and 117 symbols.
- Tic-Tac-Toe: This database encodes the complete set of possible board configurations at the end of tic-tac-toe games. There are 958 instances, 9 attributes, and 27 symbols.
- Splice: The data is used to recognize two types of splice junctions in DNA sequences: exon/intron (EI) or intron/exon (IE) sites. There are 3,190 instances, 60 attributes, and 240 symbols.
- Breast Cancer: This is the one of three domains provided by the Oncology Institute on lymphography and primary-tumor. This data set includes 296 instances, 9 attributes, and 89 symbols.
- Hayes-Roth: This data is on human subjects classification. There are 160 instances, 4 attributes, and 15 symbols.
- Monk: This is the basis of a first international comparison of learning algorithms. There are 432 instances, 6 attributes, and 17 symbols.

The statistics of these data sets are listed in Table 2.

Before reporting the classification performance, we first plot the embedding results of the categorical symbols in three data sets, namely Tic-Tac-Toe, Balance, and Splice, in Figure 1. An interesting observation is that for Tic-Tac-Toe and Balance, the number of symbols is relatively small, and symbols used to describe the same attribute (marked with the same color) tend to be close to each other. In particular, note that in the Tic-Tac-Toe game, each attribute (corresponding to one configuration) have three symbols/moves, and these symbols from the same attributes tend to be close and form a line, forming a globally (approximately) symmetric embedding. For the Splice data set, since the number of symbols is large, the grouping is insignificant.

The setup of different algorithms are as follows. For logistic regression, we tried both L_1-norm and L_2-norm regularizations; and the regularization parameter is chosen in the gird $\{0.1, 1, 10, 100, 1000\}$. For decision tree, the function used to measure the quality of a split is the *Gini* criterion, the minimum number of samples required to split an internal node is chosen from the candidate values $\{2, 4, 6, 8\}$, and the minimum number of samples required to be at a leaf node is chosen in the values $\{1, 2, 3, 4\}$. For our method, we have adopted three types of base distance measures, i.e., the cosine distance, the normalized co-occurrence, and the mutual information[4], to learn an optimal symbol distance matrix. The objective function is chosen as $J_{\mathcal{S}}^{(\theta,2)}/J_{\mathcal{D}}^{(\theta,2)}$ as specified in Eq. 4.12 with $\theta = 2$. In building the initial graph among the symbols, we only connect one symbol with top l symbols that have closest distances, where $l = \min(\frac{c}{2}, d)$, with c being the total number of symbols. The shortest path distances are then calculated on this graph.

For all the algorithms, we use 5-fold cross validation to select the best parameters if there are any. The reported results are based on the average of 30 repeats. In each repeat, 50% of the data is randomly selected for training, and the rest is used for testing. The classification accuracies are reported in Table 3. Algorithms whose performance is significantly better than others via the paired student-t test with a confidence level that is at least 95% is highlighted. As can be observed, on most (6 out of 7) data sets, our approach gives the best results. Next comes the density-based coding method.

[4]The two similarity measures are transformed to distances using the $\log(-x)$ function.

The dummy coding and decision tree algorithms perform relatively worse compared with other algorithms. This clearly demonstrates the effectiveness of our method in finding a reliable, numerical representation for categorical data analysis.

7 Conclusion and Future Work

In this paper, we propose a novel method to obtain numerical representation/embedding for categorical data. The basic idea is to learn a pairwise distance among symbols, which captures both important global proximity relations, as well as the class label information such that the resultant embedding would lead to a good generalization performance. With this transform, popular learning algorithms for numerical data can be readily applied in categorical data analysis. In the future, we will study the more challenging problem of mixed numerical and categorical data, as well as extending our method to regression tasks.

Acknowledgements

Jie Zhang is supported by the National Science Foundation of China (NSFC 61104143), and special Funds for Major State Basic Research Projects of China (2015CB856003).

References

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[2] T. Pang-Ning, V. Kumar and M. Steinbach, *Introduction to Data Mining*. Addison-Wesley Longman, 2005.

[3] Z. Chen, Y. Xie, Y. Cheng, K. Zhang, A. Agrawal, W. Liao, N. F. Samatova, and A. N. Choudhary, *Forecast Oriented Classification of Spatio-Temporal Extreme Events*, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, 2952-2954, 2013.

[4] L. Breiman, J. Friedman and R. Olshen, C. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA, 1984.

[5] T. Cox and M. Cox, *Multidimensional Scaling*, London, U.K.: Chapman & Hall, 1994.

[6] C. Cortes and V. Vapnic, *Support Vector Networks*, Machine Learning, 20, 273-297, 1995.

[7] S. Guha, R. Rastogi and K. Shim, *ROCK: A Robust Clustering Algorithm for Categorical Attributes*, Proceedings of the 15th International Conference on Data Engineering, 512 - 521, 1999.

[8] Z. Chen, K. Padmanabhan, A. Rocha, Y. Shpanskaya, J. Mihelcic, K. Scott, and N. F. Samatova, *SPICE: Discovery of Phenotype-determining Component Interplays*, BMC Systems Biology, 6(1):40+, 2012.

[9] V. Ganti, J. Gehrke and R. Ramakrishnan, *CACTUS–Clustering Categorical Data Using Summaries*, Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 73-83, 1999.

[10] J. B. Tenenbaum, V. de Silva and J. C. Langford, *A Global Geometric Framework for Nonlinear Dimensionality Reduction* , Science 290 (5500): 2319-2323, 22 December 2000.

[11] J. Cohen, P. Cohen, S.G. West and L.S. Aiken,*Applied multiple regression/correlation analysis for the behavioral sciences (3rd ed.)* , New York, NY: Routledge, 2003.

[12] Z. Chen, W. Hendrix, H. Guan, I. Tetteh, A. Choudhary, F. Semazzi, and N. Samatova, *Discovery of Extreme Events-related Communities in Contrasting Groups of Physical System Networks*, Data Mining and Knowledge Discovery, vol.27, 225-258, 2013.

[13] M. Belkin and P. Niyogi, *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation*, Neural Computation 15, 1373–1396 (2003).

[14] C. E. Rasmussen and C.K. I. Williams, *Gaussian Processes for Machine Learning*, The MIT Press, 2006.

[15] A. Agresti. *Categorical Data Analysis*, 3rd Edition, John Wiley, New York, 2012.

[16] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, San Diego: Academic Press, 2nd edition, 1990.

[17] W. Chen, Y. Chen, Y. Mao and B. Guo, *Density-based Logistic Regression*, Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 140–148, 2013.

[18] D. Gibson, J. Kleinberg and P. Raghavan, *Clustering Categorical Data: An Approach Based on Dynamical Systems*, Proceedings of the 24rd International Conference on Very Large Data Bases, 311-322, 1998.

[19] X. Huo, X. Ni and A. K. Smith, *Chapter A Survey of Manifold-Based Learning Methods*, Recent Advances in Data Mining of Enterprise Data Algorithms and Applications, 691-745, 2004.

[20] G. R. G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Barlett, and M. J. Jordan (2004). *Learning the Kernel Matrix with Semidefinite Programming*. Journal of Machine Learning Research, 5, 27-72.

[21] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. *Multiple Kernel Learning, Conic Duality, and the SMO Algorithm*. Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.

[22] A. Rakotomamonjy, F. R. Bach, and I. Rouen, *SimpleMKL*, Journal of Machine Learning Research 9 (2008) 2491-2521.

[23] S. T. Roweis and L. K. Saul. *Nonlinear Dimensionality Reduction by Locally Linear Embedding*. Science 290, 2323-2326 (2000).

[24] B. Schlköpf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[25] Q. Wang, K. Zhang, G. Jiang and I. Maric. *Improving Semi-Supervised Target Alignment via Label-Aware Base Kernels*. Proceedings of the 28th AAAI Conference on Artificial Intelligence, 2013.