Supervised Categorical Metric Learning with Schatten p-Norms

Xuhui Fan, Éric Gaussier, Yaqiong Li

Supervised Categorical Metric Learning with Schatten p-Norms

Xuhui Fan, Eric Gaussier

Abstract—Metric learning has been successful in learning new metrics adapted to numerical datasets. However, its development on categorical data still needs further exploration. In this paper, we propose a method, called CPML for *categorical projected metric learning*, that tries to efficiently (i.e. less computational time and better prediction accuracy) address the problem of metric learning in categorical data. We make use of the Value Distance Metric to represent our data and propose new distances based on this representation. We then show how to efficiently learn new metrics. We also generalize several previous regularizers through the Schatten p-norm and provides a generalization bound for it that complements the standard generalization bound for metric learning. Experimental results show that our method provides state-of-the-art results while being faster.

Keywords—*Metric Learning; Categorical Data.*

I. INTRODUCTION

Metric (or distance metric) learning represents an essential task for many machine learning problems. Relying on appropriate distance metrics can boost the performance of many learning algorithms, such as k-Nearest Neighbor, of which its success is largely depended on the distance metric of the points closest to a given point. Similarly, in k-means clustering, the shortest distance between a data point and all cluster centers also determines its cluster assignment. Several important studies have been conducted in this area, including the *information theoretic metric learning* (ITML) approach [1], the *large margin nearest neighbour* (LMNN) approach[2], or the *pseudo-metric online learning algorithm* (POLA) [3].

However, in many cases, numerical features usually come along with categorical ones that also contain discriminative information. For instance, the categorical features of educational level and marriage status represent valuable information in the credit card fraud detection problem. The standard method to deal with these categorical features is to treat them as numerical ones by transforming them into binary vectors. However, the feature number is increased in a polynomial rate. Unsupervised learning methods for categorical distances, as [4], usually rely on the simple overlapping similarity, that varies from the simple counting, through co-occurrence frequency to entropy. When label information is available, the supervised-learning of categorical measures [5], [6] is further developed. However, these methods either ignore the correlation between data samples, or come at a heavy computational

cost. In addition, none of these studies provide theoretical guarantees on the generalization bound of the learned metric.

To address the above problems, in our work, we put forward a new method, namely *categorical projected metric learning* (CPML), to efficiently learn metrics on categorical features and utilize them in real classification tasks. First, we employ the standard *value distance metric* (VDM) [7] to project each feature value into a class-based vector. Then these vectors are re-arranged to define new distances relying on the correlation between features. These new defined distances are further utilized in k-Nearest Neighbor classification tasks. Comparing to previous methods, our approach is superior in terms of computational cost, without loss of classification accuracy. It also comes with theoretical guarantees that ensure its reliability.

To achieve this, we apply the Schatten p-norm ($p \geq 1$) to regularize the eigenvalues of the metric and promote low rank solutions. Several popular regularizers are special cases of this Schatten p-norm; $p = 1$ refers to the trace norm, $p = 2$ corresponds to the Frobenius norm and $p = \infty$ represents exactly the maximum eigenvalue norm [8]. Correspondingly, we provide the generalization bound for this Schatten p-norm ($p \geq 1$), as a supplement for the standard generalization bound in metric learning literature [9].

On the experimental part, we test the performance of our model in different scenarios. By adding different number of noisy features, our model is shown to be able to correctly identify the noisy features and "denoise" them. By testing the running time in different data sizes and class numbers, we show that the class number hardly influences our model's running time. Lastly, detailed results obtained on synthetic and real world data sets confirm our models' competitive results against other benchmark models.

The remainder of this paper is organized as follows. In Section II, we introduce the *value distance metric* (VDM) method and the general framework of metric learning. We then we propose the *categorical projected metric learning* (CPML) framework aiming at efficiently learning metrics on categorical features for classification tasks. A generalization bound for the general Schatten p-norm ($p \geq 1$) is provided in Section IV. After a literature review in Section V, we provide and discuss experimental results in Section VI. The conclusion and future work can be found in Section VII.

II. PRELIMINARY KNOWLEDGE

Throughout this study, we will use the notations provided in Table I.

As is common setting, our training data includes the observations $\boldsymbol{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)$ and the corresponding labels

---

Xuhui Fan is with School of Mathematics & Statistics, University of New South Wales, Australia.: (email: xhfan.ml@gmail.com).

Eric Gaussier is with Laboratory of Informatics of Grenoble, University of Joseph Fourier, France: (email: eric.gaussier@imag.fr).

TABLE I. NOTATION TABLE

Notation	Explanation
n	Number of data points
D	Number of features
C	Number of classes
M	$D \times D$ matrix (metric to be learned)
s_d	Number of possible values for feature d
$s_{\max}$	Maximum number of possible values for all the features
$\phi(\boldsymbol{x}_i)$	VDM-based projection of example $\boldsymbol{x}_i$ ($D \times C$ matrix)
$\phi(\boldsymbol{X})$	VDM-based projection of the dataset $\boldsymbol{X}$ ($D \times C \times n$ tensor)
$N_{cd}(f)$	Number of times value f is observed for feature d in class c
$R_+^{D \times D}$	Set of semi-definite positive matrices

TABLE II. CATEGORICAL DATA EXAMPLE

ID	Occupation	Education	Marital	Risk
1	Accountant	Bachelor	Married	Low
2	Doctor	Master	Married	Low
3	Plumber	TAFE	Single	High
4	Plumber	High school	Single	Middle
5	Doctor	Master	Married	Middle
6	Accountant	Master	Single	High

$\{y_i\}_{i=1}^n$. Furthermore, $\forall i$, $\boldsymbol{x}_i$ contains D categorical features, i.e. $\boldsymbol{x}_i = [\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{iD}]^\top$, each $\boldsymbol{x}_{id}$ taking value in $\{1, \ldots, s_d\}$, where s_d denotes the number of possible values for feature d.

We now introduce the representation we use for categorical features and provide the general framework for metric learning.

A. Value Distance Metric

The *value distance metric* (VDM) is a method for representing categorical features into a C-dimensional normalized vector (C corresponds to the number of classes, see Table I). $\forall d \in \{1, \ldots, D\}$, VDM partitions the whole dataset into s_d subgroups, where the data points in the same subgroup have the same value on feature d. Then, VDM would histogram the data points according to their corresponding class labels and the histogram is normalized to represent the dth feature's class distribution.

More specifically, the feature $\boldsymbol{x}_{id}$ is first transformed into the C-length vector as follows:

$$\phi(\boldsymbol{x}_{id} = f) = [\widehat{P}(y=1|\boldsymbol{x}_{id}=f), \ldots, \widehat{P}(y=C|\boldsymbol{x}_{id}=f)]^\top \quad (1)$$

where $\widehat{P}(y=c|\boldsymbol{x}_{id}=f)$ is the estimate of the probability of having class c when the dth feature of point i has value f. It is defined as:

$$\widehat{P}(y=c|\boldsymbol{x}_{id}=f) = \frac{N_{cd}(f)}{\sum_{c=1}^C N_{cd}(f)} \quad (2)$$

Here $N_{cd}(f) = \sum_{i=1}^N \mathbf{1}(x_{id}=f, y_i=c)$ denotes the number of times the feature value f occurs in class c for the dth feature, and $\sum_{c=1}^C N_{cd}(f) = \sum_{i=1}^N \mathbf{1}(x_{id}=f)$ refers to the total appearances of feature value f for the dth feature. VDM is thus a class-based projection, inspired from the original value distance metric [7], [5]. Recent work in [6] has also employed this projection.

We take the credit risk data as an example. In Table II, we have a set of 6 persons. The occupations, educations and marital status of these persons are taken as features, and the credit risk level is taken as labels. For person 1's occupation feature, which is Accountant, we can first estimate

the probability of different credit risk for person being the accountant as follows:

$$\widehat{P}(y = \text{Low}|\boldsymbol{x}_{11} = \text{Accountant})$$
$$= \frac{\sum_{i=1}^6 \mathbf{1}(y_i = \text{Low}, \boldsymbol{x}_{i1} = \text{Accountant})}{\sum_{i=1}^6 \mathbf{1}(\boldsymbol{x}_{i1} = \text{Accountant})} = \frac{1}{2}$$

Correspondingly, we can represent Eq. (1) as:

$$\phi(\boldsymbol{x}_{i1} = \text{Accountant}) = (\frac{1}{2}, 0, \frac{1}{2}) \quad (3)$$

B. Metric Learning

Metric learning naturally arises in the question of how to assess the similarity of different objects. Its corresponding distance function is usually set as the Mahalanobis distance, with the inverse covariance matrix as the unknown variables. With the prior knowledge of class labels or side information, we are trying to find an optimal metric that aims at minimizing the number of errors made.

$$\arg \min_M f(M) + \lambda r(M)$$

Here $f(\cdot)$ is the loss function, $r(\cdot)$ is the regularization function and λ is the tuning parameter balancing the loss incurred and the model complexity.

Our approach, as well as most metric learning approaches, fits within this general setting.

III. CATEGORICAL METRIC LEARNING

VDM transforms each data point $\boldsymbol{x}_i$ into a $D \times C$ matrix, denoted as $\phi(\boldsymbol{x}_i) \in \mathbb{R}^{D \times C}, \forall i \in \{1, \ldots, n\}$. Each row in $\phi(\boldsymbol{x}_i)$ is an estimation of the class distribution, so that $\phi(\boldsymbol{x}_i) \cdot \mathbf{1}_{C \times 1} = \mathbf{1}_{D \times 1}$. The cth column, $\phi_c(\boldsymbol{x}_i)$, in $\phi(\boldsymbol{x}_i)$ represents the popularity of class c in different features, which could be denoted as $\phi(\boldsymbol{x}_i) = [\phi_1(\boldsymbol{x}_i), \phi_2(\boldsymbol{x}_i), \ldots, \phi_C(\boldsymbol{x}_i)]$. The whole projected dataset is $\phi(\boldsymbol{X}) \in \mathbb{R}^{D \times C \times n}$.

Based on this representation, we define two distances that take into account the correlations between features inside each class. They differ in the way of treating the metric learned for different classes (different metrics for different classes are learned in one case, whereas a single metric for all classes is learned in the other case).

Definition 1. CPm: *The categorical projected multi (CPm) distance considers the features' individual metrics $\{M_c\}_{c=1}^C$*

among different classes and is defined by:

$$
\begin{aligned}
&d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
&= \sum_{c=1}^{C} Tr\left((\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))(\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))^\top M_c^\top\right) \\
&= \sum_{c=1}^{C} Tr\left(A_c^{ij} M_c^\top\right) = \sum_{c=1}^{C} \sum_{p,q} A_{c,pq}^{ij} M_{c,pq}
\end{aligned}
\tag{4}
$$

Here $A_c^{ij} = (\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))(\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))^\top, M_c \in \mathbb{R}_+^{D \times D}$.

Definition 2. CPs: *The categorical projected single (CPs) distance considers the features' correlation by assuming that all the classes share the same metric, and is defined by:*

$$
\begin{aligned}
&d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
&= Tr\left((\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))^\top M^\top\right) \\
&= Tr(A^{ij} M^\top) = \sum_{p,q} A_{pq}^{ij} M_{pq}
\end{aligned}
\tag{5}
$$

Here $A^{ij} = (\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j))^\top, M \in \mathbb{R}_+^{D \times D}$.

We assume the positive semi-definite property of $\{M_c\}_{c=1}^{C}$ and M to ensure the the non-negativeness of $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The following property (see the proof in the appendix) furthermore shows that the above definitions correspond to valid distances.

Property 1. *Assume $M \succeq 0$, then $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$, and, $\forall \boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k$, we have $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq d_M(\boldsymbol{x}_i, \boldsymbol{x}_k) + d_M(\boldsymbol{x}_k, \boldsymbol{x}_j)$.*

It is obvious $d_M(\boldsymbol{x}_i, \boldsymbol{x}_i) = 0$ and $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_M(\boldsymbol{x}_j, \boldsymbol{x}_i)$. Thus, our definitions of CPm and CPs $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$ in Eq. (4) and (5) correspond to valid metrics.

A. Objective Function & Optimization

From the class information in the data set, one can derive a constraint set based on triplets of points, $\mathcal{T} = \{(i, j, k) | d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) < d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)\}$, that indicates that any point should be closer to points of the same class than to points of other classes. From this constraint set, our task is to find an optimal M such that the *empirical loss $\varepsilon_{\mathcal{T}}(M)$ minimized*. The empirical loss $\varepsilon_{\mathcal{T}}(M)$ is here defined as:

$$
\varepsilon_{\mathcal{T}}(M) = \frac{1}{|\mathcal{T}|} \sum_{(i,j,k) \in \mathcal{T}} [d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b - d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)]_+
$$

$[\cdot]_+$ is the hinge loss function, and b is the margin parameter, often set to $b = 1$.

The above setting can be used for constraint sets based on pairs of points, $\mathcal{P} = \{(i, j) | d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) < b\}$, in which case the empirical loss takes the form:

$$
\varepsilon_{\mathcal{P}}(M) = \frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} [1 + d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) - b]_+
$$

It can also be used for quadratic constraint sets, based on 4-uples of data points, $\mathcal{Q} = \{(i, j, k, l) | d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) <$

$d_M(\boldsymbol{x}_k, \boldsymbol{x}_l)\}$, in which case the empirical loss takes the form:

$$
\varepsilon_{\mathcal{Q}}(M) = \frac{1}{|\mathcal{Q}|} \sum_{(i,j,k,l) \in \mathcal{Q}} [d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b - d_M(\boldsymbol{x}_k, \boldsymbol{x}_l)]_+
$$

In real world applications, no single constraint would dominate the other two in different scenarios. When two points from different classes can indeed be very close or far away to each other, the triplet constraint set $\mathcal{T}$ and the pair constraint set $\mathcal{P}$ do not make strong assumptions in this case, whereas the quadratic constraint $\mathcal{Q}$ might set these assumptions. We believe that both $\mathcal{P}$ and $\mathcal{T}$ define valuable information on which to learn a new metric. We present here the solution of the optimization problem based on $\mathcal{T}$, but the same development can be used for $\mathcal{P}$.

Algorithm 1 The CPML framework

Input: $\phi(\boldsymbol{X})$: $D \times C \times n$ projected tensor for the whole dataset;
α_t, stepsize at the tth iteration; α, stepsize's decreasing rate
λ: regularization parameter; $\mathcal{T}$: constraint set for the whole dataset
Output: the resulted metrics M
Initialize $M = I_{D \times D}$
Compute A valuees between each data pair
repeat
    Compute the violated constraint set $\widehat{\mathcal{T}}$
    Compute the subgradient for f:

$$
\frac{\partial}{\partial M} f(M) = \frac{\partial r}{M}(M) + \frac{\lambda}{|\mathcal{T}|} \sum_{(i,j,k) \in \widehat{\mathcal{T}}} \left[A^{ij} - A^{ik}\right] \tag{6}
$$

    Use **backtrack line search** method to determine α_t';
    Update M as: $M_{t+1} = M_t - \alpha_t \frac{\partial}{\partial M} f(M)$;
    Project M back into the Positive Semi-Definite cone
until converge

By choosing a suitable metric regularizer $r(M)$, our problem amounts to minimize the objective function $f(M)$:

$$
\arg \min_M f(M) = \arg \min_M \{\varepsilon_{\boldsymbol{x}}(M) + \lambda \cdot r(M)\}
$$

where λ is the regularization parameter.

The choice of the metric regularizer affects the structure of the solution learned. For instance, the L_1-norm promotes sparse metric, while the trace norm encourages metrics with low rank; the Frobenius norm ($\|M\|_2^2 = \sum_{pq} M_{pq}^2$), on the other hand, tends to yield robust solutions.

If $r(M)$ is a convex function, the objective function $f(M)$ is also convex with respect to M. As the empirical hinge loss is non-differentiable at 0, we apply the Projected Subgradient Descent method to seek the optimal value of M. The subgradient of $f(M)$ is composed of two terms: one in the regularization, i.e. the matrix $\frac{\partial r}{M}(M)$, and the other in the empirical loss function. The subgradient of the empirical loss is the sum of the sub gradient of the hinge loss in each constraint in $\mathcal{T}$. For each $(i, j, k) \in \mathcal{T}$, the subgradient direction is

0 if the loss is 0, i.e. $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b \leq d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)$. For $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b > d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)$, we get:

$$\frac{\partial \left[d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b - d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)\right]_+}{\partial M}$$
$$=\frac{\partial \left[Tr(A^{ij}M) + b - Tr(A^{ik}M)\right]_+}{\partial M}$$
$$=A^{ij} - A^{ik}$$

Algorithm 2 Backtrack line search for determining the stepsize [10]

Input: $\phi(\boldsymbol{X})$, M: projected matrix, current metric
f: objective function
$\frac{\partial}{\partial M} f(M)$: gradient of the objective function f
α: stepsize decreasing rate, usually set to 0.1
Output: the stepsize α_t
Initialize $\alpha_t = 1$;
repeat
$\alpha_t = \alpha_t \cdot \alpha$;
until $f(M - \alpha_t \frac{\partial}{\partial M} f(M)) \leq f(M) - \frac{\alpha_t}{2} \|\frac{\partial}{\partial M} f(M)\|^2$

Thus, the subgradient of $f(M)$ is:

$$\frac{\partial}{\partial M} f(M) = \frac{\partial r}{M}(M) + \frac{\lambda}{|\mathcal{T}|} \sum_{(i,j,k) \in \widehat{\mathcal{T}}} \left[A^{ij} - A^{ik}\right]$$

where $\widehat{\mathcal{T}} = \{(i,j,k) | d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + b > d_M(\boldsymbol{x}_i, \boldsymbol{x}_k)\}$ denotes the set of triplets for which the constraint is violated.

After each gradient step $M_{t+1} = M_t - \alpha_t \frac{\partial}{\partial M} f(M_t)$, we need to project M back to the positive semi-definite cone. This is conducted by setting the negative eigenvalues in M_t to be 0.

The complete process is described in Algorithm 1. It is important to note that the A^{ij} values can be computed before the iteration, which reduces the computational cost.

IV. RADEMACHER COMPLEXITY AND SCHATTEN p-NORMS

In choosing the regularizer $r(M)$, we here rely on a Schatten p-norm of the learned metric $[Tr(M^p)]^{\frac{1}{p}}, p \geq 1$, that generalizes several well-known metric-regularizers. The study in [9] gives a generalization bound on the pair-comparison empirical loss, defined by:

$$\varepsilon_{\mathcal{T}}(M, b) = \frac{1}{n(n-1)} \sum_{i \neq j} [1 + r_{ij}(d_M(x_i, x_j) - b)]_+$$

Here $r_{ij} = 1$ if $y_i = y_j$, otherwise $r_{ij} = 1$; b is the margin parameter. One expects that $r_{ij} = 1$ if $d_M(x_i, x_j) \leq b$, and $r_{ij} = 0$ otherwise.

Thus, the objective function to be minimized is:

$$\frac{1}{n(n-1)} \sum_{i \neq j} [1 + r_{ij}(d_M(x_i, x_j) - b)]_+ + \lambda \|M\|^2$$

where $\|M\|^2$ is the metric regularizer, and $\lambda > 0$ the regularization parameter.

From this, the study in [9] derives the following generalization bound, that holds with probability at least $(1 - \delta)$:

$$\varepsilon(M, b) - \varepsilon_{\mathcal{T}}(M, b) \leq \frac{4R_n(M)}{\sqrt{\lambda}} + \frac{4(3 + 2X_*/\sqrt{\lambda})}{\sqrt{n}}$$
$$+ 2(1 + X_*/\sqrt{\lambda}) \left(\frac{2\ln\left(\frac{1}{\delta}\right)}{n}\right)^{\frac{1}{2}}$$

where $\widehat{R}_n(M)$ denotes the empirical Rademacher complexity, defined as:

$$\widehat{R}_n(M) = \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_\sigma \| \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \sigma_i X_{i(\lfloor \frac{n}{2} \rfloor + i)} \|_*$$

and $X_* = \sup_{x, x' \in \mathcal{X}} \|(x - x')(x - x')^\top\|_*$ (X_* measures the diameter of the domain of $\mathcal{X}$). $\| \cdot \|_*$ denotes the dual norm for a given norm $\| \cdot \|$.

However, the study in [9] does not consider the Schatten p-norm as a metric regularizer, and we provide here a bound on the Rademacher complexity of the Schatten p-norm regularizer for the case $p \geq 1$. This bound complements the study in [9] for the pair-comparison case. To do this, we first define the expectation of the empirical Rademacher complexity:

$$R(M) = \mathbb{E}_{\boldsymbol{z}} \widehat{R}_n(M) = \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z}, \sigma} \| \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \sigma_i X_{i(\lfloor \frac{n}{2} \rfloor + i)} \|_*$$

Then, we have the following theorem.

Theorem 1. *The Rademacher Complexity of our distances in the Schatten p-norm ($p \geq 1$) in the pair-comparison empirical loss case is bounded by:*

$$R(M) \leq \begin{cases} D^{\frac{1}{2} - \frac{1}{p}} \cdot \frac{2X_*}{\sqrt{n}}, & 2 \leq p; \\ D^{1 - \frac{1}{p}} \cdot \frac{2X_*}{\sqrt{n}}, & 1 \leq p < 2. \end{cases} \quad (7)$$

Proof: The dual norm of the Schatten p-norm is the Schatten q-norm, where q satisfies $\frac{1}{p} + \frac{1}{q} = 1$. Let us first assume that $p \geq 2$; then $1 \leq q \leq 2$. Assume $\lambda_{1,\sigma,i}, \ldots, \lambda_{d,\sigma,i}$ are the eigenvalues of the matrix $\sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \sigma_i X_{i(\lfloor \frac{n}{2} \rfloor + i)}$, we have

$$R(M) = \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z}, \sigma} \| \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \sigma_i X_{i(\lfloor \frac{n}{2} \rfloor + i)} \|_q$$
$$= \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z}, \sigma} [\sum_{k=1}^{D} (\lambda_{k,\sigma,i})^q]^{\frac{1}{q}}$$
$$\overset{H\ddot{o}lder}{\leq} \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z}, \sigma} \left\{ \left[\sum_{k=1}^{D} ((\lambda_{k,\sigma,i})^q)^{\frac{2}{q}}\right]^{\frac{q}{2}} \cdot D^{\frac{2-q}{2}} \right\}^{\frac{1}{q}} \quad (8)$$
$$= \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z}, \sigma} \left[\sum_{k=1}^{D} (\lambda_{k,\sigma,i})^2\right]^{\frac{1}{2}} \cdot D^{\frac{2-q}{2q}}$$
$$\overset{[9]}{\leq} D^{\frac{2-q}{2q}} \cdot \frac{2X_*}{\sqrt{n}}$$

Let us now assume that $1 \leq p < 2$; then $q > 2$. Let $\lambda_{\max,\sigma,i} = \max\{\lambda_{1,\sigma,i}, \ldots, \lambda_{d,\sigma,i}\}$.

$$
\begin{aligned}
R(M) &= \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z},\sigma} [\sum_{k=1}^{d} (\lambda_{k,\sigma,i})^q]^{\frac{1}{q}} \leq \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z},\sigma} [d^{\frac{1}{q}} \cdot \lambda_{\max,\sigma,i}] \\
&= d^{\frac{1}{q}} \cdot \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z},\sigma} [\lambda_{\max,\sigma,i}] \\
&\leq d^{\frac{1}{q}} \cdot \frac{1}{\lfloor \frac{n}{2} \rfloor} \mathbb{E}_{\boldsymbol{z},\sigma} \left[\sum_{k=1}^{d} (\lambda_{k,\sigma,i})^2 \right]^{\frac{1}{2}} \\
&\overset{[9]}{\leq} d^{\frac{1}{q}} \cdot \frac{2X_*}{\sqrt{n}}
\end{aligned}
$$

(9)

It has to be noted here that, as we are using the union bound in this case, the result in the $1 \leq p < 2$ case may be loose.

The conclusion is finally obtained by summarizing Equations (8) and (9). $\square$ $\blacksquare$

Armed with this result, we can now state a generalization bound for with Schatten p-norm in the pair-comparison case.

Theorem 2. $\forall 0 < \delta < 1$, with probability $(1 - \delta)$, we have that

$$
\varepsilon(M,b) - \varepsilon_{\mathcal{T}}(M,b) \leq \frac{8 D^{\alpha(p)-\frac{1}{p}} X_*}{\sqrt{n\lambda}} + \frac{4(3 + 2X_*/\sqrt{\lambda})}{\sqrt{n}} \\
+ 2(1 + X_*/\sqrt{\lambda}) \left(\frac{2 \ln \left(\frac{1}{\delta} \right)}{n} \right)^{\frac{1}{2}}
$$

Here $\alpha(p) = \begin{cases} \frac{1}{2}, & 2 \leq p; \\ 1, & 1 \leq p < 2. \end{cases}$

V. RELATED WORK

A. Metric Learning

From the seminal work of [11], the majority of studies in metric learning focuses on numerical data. An optimal metric is usually learned, from some labeled information, in the family of Mahalanobis distances. Several metric learning methods have lead to classifiers significantly better than the one based on standard metrics as the Euclidean distance (i.e. without learning). Among such methods we can cite the *large margin nearest neighbour* (LMNN) approach[2] that first determines the nearest neighbors of each point in the Euclidean space, then tries to move the point closer to its neighbors of the same classe while pushing it away from neighbors of other classes. The *information theoretical metric learning* (ITML) [1] tries to minimize the relative entropy between two multivariate Gaussian distributions under distance constraints. The *maximum-eigenvalue metric learning* [8], [12] uses the popular Frank-Wolfe algorithm to formulate the problem as a constrained maximum-eigenvalue problem, which avoids the computation of the full eigen-decomposition in each iteration.

Metric learning on numerical data usually involves a linear transformation of the original Euclidean space. Some studies, however, rely on non-linear transformations, as the χ^2-LMNN

and GB-LMNN approaches [13] that respectively make use of the χ^2 distance and regression trees. Other non-linear transformations involve the use of kernels [14], [15]. *Hamming distance metric learning* [16] has recently been proposed to learn a mapping from real-valued inputs into binary ones, with which the hash function can fully utilized to enable large scalability.

On learning the distance between categorical data, [4] have conducted an extensive comparison between various unsupervised measures. The closest work to ours is [5], which is to measure correlation structure among each feature, and [6], which considers the full class-features' correlation structure. However, the first neglects the potential correlations between the features and both of their scalability are questionable, as they individually optimize $d \cdot n_d^2$ and $(D \cdot C) \times (D \cdot C)$ matrixes.

We notice there are two algorithms proposed recently to learn metric on categorical data. The heterogeneous metric learning with hierarchical couplings (HELIC) [17] mainly focuses on the attribute-to-value and attribute coupling framework. Their insufficient utilization on the labels leads to degraded performance. Also, the DM3 method [18] considers only the frequency information within the attributes. The lack of class label incorporation makes the performance unappealing again. We show this in the experimental part.

On the theoretical generalization guarantees, [19] uses the uniform stability concept to firstly bound the deviation from *true risk* to *empirical risk*, under the Frobenious norm case. [20] gives the generalization bound without regularizers, with strong assumptions on the points' distribution. [21] has shown that robustness [22] is necessary and sufficient to be generalized well. [9] uses the notion of Rademacher complexity to derive the generalization bound for several regularizers. Our generalization bound on the Schatten p-norm ($p \geq 1$) is a supplement to this work.

B. Distance between categorical data

In addition to the VDM method we have used, there are other existing methods to quantify the distance between categorical data in the literature. Among all those methods, Hamming distance [23] is widely known for its intuitive understanding and simplicity to implementation. The idea is to treat the same value in the categorical data as 1, and 0 otherwise. However, the Hamming distance lacks the ability to model dependence within the features and also the potential connection to the class information.

The association-based metric proposed by [24] uses an indirect probabilistic method. Particularly, the metric is estimated by the sum of distances between conditional probability density function (cpdf) of other attributes given these two values, i.e., $D(x_{id}, x_{jd}) = \sum_{d'} \psi(\text{cpdf}(X|x_{id}) + \text{cpdf}(X|x_{jd}))$, where $\psi(\cdot, \cdot)$ is the distance function between two probability density functions and can be used as the Kullback-Leibler divergence. However, the distance is 0 if all the attributes are independent of each other.

The context-based metric [25], [26] is determined by a measure of symmetrical uncertainty (SU) between pairwise attributes. Particularly, SU is calculated as $\text{SU}(X_{d_1}, X_{d_2}) =$

$$\begin{pmatrix} M_{11} & M_{12} & \ldots & M_{1C} \\ M_{12} & M_{22} & \ldots & M_{2C} \\ \vdots & \vdots & \ddots & \vdots \\ M_{1C} & M_{2C} & \ldots & M_{CC} \end{pmatrix} \begin{pmatrix} M_{11} & 0 & \ldots & 0 \\ 0 & M_{22} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & M_{CC} \end{pmatrix} \begin{pmatrix} M_{11} & 0 & \ldots & 0 \\ 0 & M_{11} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & M_{11} \end{pmatrix} \begin{pmatrix} I_{D \times D} & 0 & \ldots & 0 \\ 0 & I_{D \times D} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & I_{D \times D} \end{pmatrix}$$

Fig. 1. Features' correlation structure (from left to right: KDML, CPm, CPs, Euclidean).

$2\frac{\text{IG}(X_{d_1}|X_{d_2})}{H(X_{d_1})+H(X_{d_2})}$, where $H(X_{d_1})$ is the entropy of attribute d_1 and $\text{IG}(X_{d_1}|X_{d_2})$ is the information gain. The metric between two attribute value is determined by further usage of SU. Similar as the association-based metric, the context-based metric can not work well when these attributes are independent.

Fig. 1 displays four different feature correlation structures. As we can see, the CP-m and CP-s distances are in between the full correlation considered in KDML and the simple one used in the Euclidean distance.

C. Computational complexity

The compuational complexity of our captegorical project method is determined by two parts: calculating the VDM projection and the metric learning part. In calculating the VDM projection, we calculate the corresponding projection of all the D categorical features of n nodes. For each feature value, we use the corresponding class information. Thus, the total computational cost is $\mathcal{O}(nDCs_{\max})$, where $s_{\max}$ is the maximum number of values in one feature.

In calculating the computational complexity of metric learning, we taken the matrix of A to be fixed as it is calculated in advance. Thus, for a given iteration number L and length of contraint set T, the computational time scales to $\mathcal{O}(LTCD^3)$ for CPML-m (i.e. metric learning with categorical projected multi-distance) method (and $\mathcal{O}(LTD^3)$ for CPML-s (i.e. metric learning with categorical projected single-distance) method). Here the term of $\mathcal{O}(D^3)$ refers to the spectral decomposition of M such that we can manually make it positive semi-definite. In most of the other approaches (e.g. KDML), the computation complexity would scales to $\mathcal{O}(C^3D^3)$, which is almost infeasible when the number of class is large.

As a result, the computational complexity for the CPML-m and CPML-s method can be summarized as $\mathcal{O}(nDCs_{\max} + LTCD^3)$ and $\mathcal{O}(nDCs_{\max} + LTD^3)$.

VI. EXPERIMENTS

The performance of our CPML framework is validated by experiments on synthetic dataset as well as real-world datasets. On the synthetic dataset, we mainly test the properties of CPML, e.g., the influence of the number of features, the presence of noisy features, and the running time. The real-world datasets are mainly used for evaluating the performance of different approaches. Particularly, we individually implement three baseline methods, i.e. LMNN, KDML and DM3, to the best of our understanding. For HELIC, we use the authors' kindly provided implementation.

Further, we use *triplet comparison accuracy* and *classification accuracy* to assess the performance of the methods. The triplet constraints are built by considering that any pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ from the same class should have a lower distance than any pair $(\boldsymbol{x}_i, \boldsymbol{x}_k)$ from different classes. For the *triplet comparison accuracy*, we also construct these triplets on the test data and evaluate the proportion correctly predicted. For the *classification accuracy*, we use Nearest Neighbor classification as the default classifier. Further, we randomly divide the data into 10 parts and set the ratio of training:validation:testing as $6:2:2$. The trade-off parameter λ is set as ranging from 10^{-4} to 10^4. For each scenario, the experiments are run 50 times and averaged; the summary statistics (mean, standard deviation) are reported.

A. Synthetic data

The synthetic data is generated as follows. For each categorical feature, their values are generated by a multinomial distribution, which is parameterized by uniform random variables. To distinguish different classe, we manually add a weight to one of the component for each class. After the parameter normalization, the generated feature values can ensure favored values in each feature for different classes. Thus, a total of D multinomial distributions are demanded for this generation.

More specifically, we set the size of the dataset as 1000. The number of features varies in $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 25]$, and the weights are chosen sequentially in $[0.1, 0.3, 0.5, 0.7, 0.9]$.

1) Impact of the Number of Features and of the Weights: Fig. 2 and Fig. VI shows the triplet comparison accuracy and pair-comparison classification accuracy for our methods of CPML-s and CPML-m. In these comparisons, we test the cases with 4 classes and 20 classes and use the trace-norm as the regularizer in this case, which is our Schatten 1-norm.

From these figures, we can easily see that, in general, the performance (i.e. triplet comparison accuracy and pair-comparison classification accuracy) improves when the number of features increases. This phenomenon coincides with our common knowledge that the larger the number of informative features we have, the better performance will be. On the weight comparison, it is also clear that the performance will be better with larger weight values.

These four figures also show that CPML-s and CPML-m have similar performance (we did not find any significant differences between the two models). In summary, the performance tends to be stable when the number of features lies between 10 and 15. In that case, one obtains satisfying results even on the least informative features (i.e. weight = 0.1). For
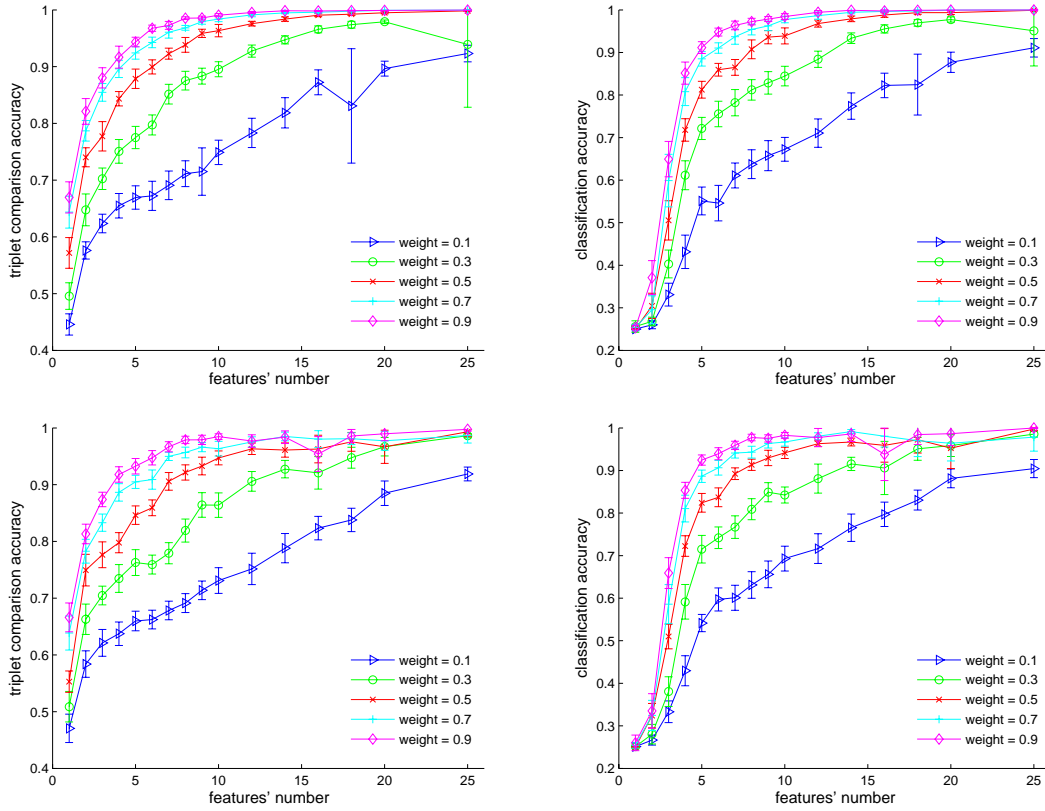
Fig. 2. Synthetic data classification performance (with 4 classes). (Top: CPML-s; bottom: CPML-m.)

the cases with 4 classes and 20 classes, we can see they have similar performance trends. The score of the latter looks to be a bit degraded, this might due to the larger number of classes.

2) Impact of Noisy Features: We assess here whether the presence of noisy features impact the valid calculation of distance. We are using the CPML-s as the exploratory method, and the norms include the trace norm $Tr(M)$, the Frobenius Norm $Tr(M^2)$ and the Schatten 3-norm $Tr(M^3)$. 8 informative features, with favored weight 0.3, are used here with a set of noisy features (# N.F. denotes the number of noisy features). We then compute the ratio between the norm of the metric on noisy feature and the norm of the metric on the whole feature space, i.e. $\frac{r(M_{N.F.})}{r(M)}$. The results are shown in Table III.

TABLE III. NOISY FEATURE'S RATIO COMPARISON (MEAN $\pm$ STANDARD DEVIATION)

# N.F.	Trace	Frobenius	L_3
1	0.006 ± 0.006	0.006 ± 0.005	0.004 ± 0.003
4	0.012 ± 0.006	0.010 ± 0.014	0.017 ± 0.015
7	0.011 ± 0.006	0.002 ± 0.016	0.017 ± 0.025
10	0.002 ± 0.006	0.001 ± 0.024	0.005 ± 0.014
13	0.017 ± 0.006	0.016 ± 0.092	0.006 ± 0.070

From Table III, we can see that, even if the number of noisy features is much larger than the number of meaningful features (13 noisy features and 8 meaningful features), our CPML-s method with various Schatten p-norms can success-

fully control the noisy features as their influence on the metric does not exceed 1.7%. This noisy feature resistance property may be explained as the result of our simple distance definitions.

3) Running Time Comparison: Fig. 4 displays comparisons on the logarithm of the running time over different methods. The left part shows the performance of CPML-s, CPML-m and KDML in terms of different classes. As we can see, our CPML-s model is the fastest one when compared to CPML-m and KDML. Also, the running time of CPML-s does not show a significant difference on the different choices for the number of classes. In contrast, both of CPML-m and KDML require heavier computation, and their running time depends on the number of classes. An interesting observation is that KDML is faster than CPML-m when the number of classes is small. The right part shows the performance of CPML-s, CPML-m, KDML, LMNN, HELIC and POLA when the number of classes are set to 12. Due to the online learning nature, POLA is the fastest to obtain the result as it only needs to scan the whole dataset once. Among all other comparison methods, CPML-s require the smallest running time. When HELIC requires smaller running time than the CPML-m algorithm, KDML and LMNN usually require more running time than the CPML-m algorithm (especially the size of dataset is large).
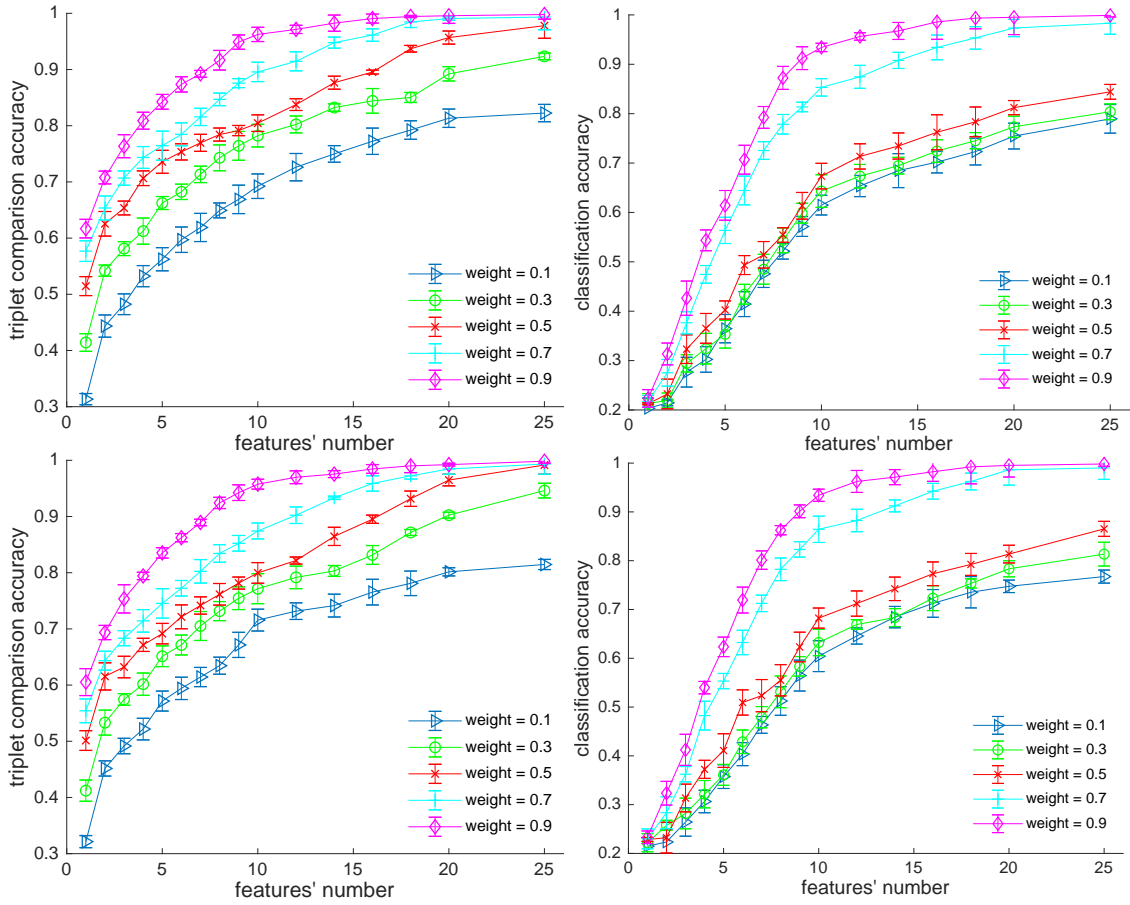
Fig. 3.   Synthetic data classification performance (with 20 classes). (Top: CPML-s; bottom: CPML-m.)

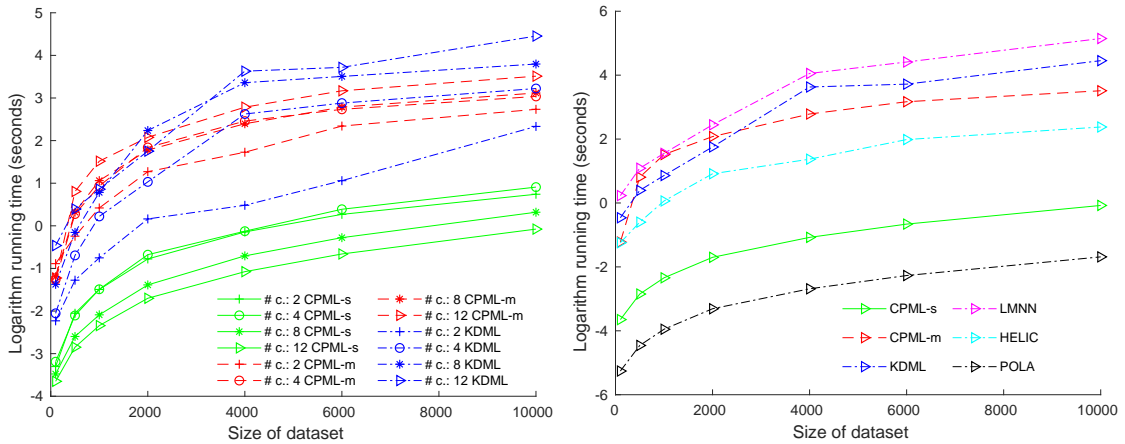Fig. 4.   Left: logarithm running time comparison of CPML-s, CPML-m and KDML with respect to different number of classes. Here # c. denotes the number of classes. The blue lines denote the required running time for KDML, the red lines is for our CPML-m and the green lines CPML-s; right: logarithm running time comparison of CPML-s, CPML-m, KDML, LMNN, HELIC and POLA with respect to different sizes of datasets.

B. Real world datasets [27]

We select 23 real world datasets to test the performance of the CPM framework: Car, Balance, Mushroom, Voting, Nursery, Monks1, Monks2, Tic-tac-toe, Krkopt, Adult, Connect4, Census, Zoo, DNAPromoter, Lymphography, Audiology, Housevotes, Spect, Soybeanlarge, DNANomial, Splice, Krvskp and Led24. The detail information of these 23 datasets, including number of instances (# I.), number of categorical features (# C.F.) and number of classes (# C.), is shown in Table IV.

TABLE IV. UCI DATASETS' DETAIL INFORMATION

Dataset	# I.	# C.F.	# C.
Balance	625	4	3
Car	1,728	6	4
Krkopt	28,056	6	17
Monks2	12,958	8	4
Tic-tac-toe	958	8	4
Adult	30,162	8	2
Nursery	12,960	8	4
Zoo	101	16	7
Housevotes	232	16	2
Voting	435	16	2
Monks1	435	16	2
Lymphography	148	18	4
Mushroom	8,124	22	2
Spect	267	22	3
Led24	3,200	24	10
Soybeanlarge	307	35	19
Census	299,285	35	2
Krvskp	3,196	36	2
Connect4	67,557	42	3
DNAPromoter	106	57	2
DNANominal	3,186	60	4
Splice	3,190	60	4
Audiology	226	69	24

As shown by the results in Table VI-B and Table VI-B, our methods achieve very competitive performances against other baseline methods. In all the 23 datasets, the CPML methods obtain the best performance in most cases. Even if in some datasets like Voting they may not be the best, their performance is quite close to it. What is more, their performance is consistent among different datasets. For other comparison methods, the HELIC and DM3 usually perform better than the KDML and LMNN. This might due to that CPML, HELIC and DM3 are specially designed for categorical data.

VII. CONCLUSION

Through the experimental validation, we have seen that our models provide competitive and robust classification results compared to previous ones. At the same time, the required computation time is reduced significantly, especially for our CPML-s method.

There are many future work that can be done based on this framework. The number of values for each feature, as well as the distribution of these values, influence the distance calculation and this should be considered. Moreover, our learning task fits within the supervised learning scenario, where all the data points label are known before hand. In practice, however, one sometimes has access to only side-information concerning labels, which corresponds to a semi-supervised learning scenario. A new learning design is needed for this scenario, that requires defining a new projection function.

REFERENCES

[1] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proceedings of the 24th International Conference on Machine learning*. ACM, 2007, pp. 209–216.

[2] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *Advances in Neural Information Processing Systems*, vol. 18, p. 1473, 2006.

[3] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng, "Online and batch learning of pseudo-metrics," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 94.

[4] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in *In Proceedings of the eighth SIAM International Conference on Data Mining*, pp. 243–254.

[5] V. Cheng, C.-H. Li, J. T. Kwok, and C.-K. Li, "Dissimilarity learning for nominal data," *Pattern Recognition*, vol. 37, no. 7, pp. 1471 – 1477, 2004.

[6] Y. He, W. Chen, Y. Chen, and Y. Mao, "Kernel density metric learning," in *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, Dec 2013, pp. 271–280.

[7] C. Stanfill and D. Waltz, "Toward memory-based reasoning," *Communications of the ACM*, vol. 29, no. 12, pp. 1213–1228, 1986.

[8] Y. Ying and P. Li, "Distance metric learning with eigenvalue optimization," *The Journal of Machine Learning Research*, vol. 13, pp. 1–26, 2012.

[9] Q. Cao, Z.-C. Guo, and Y. Ying, "Generalization bounds for metric and similarity learning," *CoRR*, vol. abs/1207.5437, 2012.

[10] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[11] E. P. Xing, M. I. Jordan, S. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information," in *Advances in Neural Information Processing Systems*, 2002, pp. 505–512.

[12] J. Yu, X. Yang, F. Gao, and D. Tao, "Deep multimodal distance metric learning using click constraints for image ranking," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–11, 2016.

[13] D. Kedem, S. Tyree, F. Sha, G. R. Lanckriet, and K. Q. Weinberger, "Non-linear metric learning," in *Advances in Neural Information Processing Systems*, 2012, pp. 2573–2581.

[14] A. Globerson and S. T. Roweis, "Visualizing pairwise similarity via semidefinite programming," in *International Conference on Artificial Intelligence and Statistics*, 2007, pp. 139–146.

[15] L. Torresani and K.-c. Lee, "Large margin component analysis," in *Advances in Neural Information Processing Systems*, 2006, pp. 1385–1392.

[16] M. Norouzi, D. J. Fleet, and R. Salakhutdinov, "Hamming distance metric learning." in *NIPS*, 2012, pp. 1070–1078.

[17] C. Zhu, L. Cao, Q. Liu, J. Yin, and V. Kumar, "Heterogeneous metric learning of categorical data with hierarchical couplings," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 7, pp. 1254–1267, 2018.

[18] H. Jia, Y. Cheung, and J. Liu, "A new distance metric for unsupervised learning of categorical data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 5, pp. 1065–1079, 2016.

[19] R. Jin, S. Wang, and Y. Zhou, "Regularized distance metric learning: Theory and algorithm," in *Advances in Neural Information Processing Systems*, 2009, pp. 862–870.

[20] W. Bian and D. Tao, "Learning a distance metric by empirical loss minimization," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, no. 1, 2011, p. 1186.

[21] A. Bellet and A. Habrard, "Robustness and generalization for metric learning," *arXiv preprint arXiv:1209.1086*, 2012.

[22] H. Xu and S. Mannor, "Robustness and generalization," *Machine learning*, vol. 86, no. 3, pp. 391–423, 2012.

[23] R. W. Hamming, "Error detecting and error correcting codes," *The Bell system technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

TABLE V. DIFFERENT MODELS' PERFORMANCE ON **triplet comparison** ACCURACY (MEAN ± STANDARD DEVIATION)

Dataset	Euclidean	CPML-single	CPML-multi	KDML	LMNN	HELIC	DM3
Car	0.593 ± 0.020	0.619 ± 0.023	**0.624 ± 0.019**	0.579 ± 0.022	0.579 ± 0.013	0.617 ± 0.019	0.609 ± 0.008
Balance	0.693 ± 0.033	0.848 ± 0.025	0.843 ± 0.016	0.851 ± 0.023	0.849 ± 0.003	**0.853 ± 0.017**	0.839 ± 0.021
Mushroom	0.864 ± 0.004	**0.894 ± 0.004**	0.877 ± 0.013	0.879 ± 0.004	0.891 ± 0.004	0.889 ± 0.008	0.864 ± 0.007
Voting	0.894 ± 0.041	**0.934 ± 0.040**	0.929 ± 0.031	0.929 ± 0.060	0.926 ± 0.003	0.909 ± 0.010	0.910 ± 0.007
Nursery	0.897 ± 0.004	**0.901 ± 0.005**	0.898 ± 0.004	0.894 ± 0.006	0.890 ± 0.006	0.878 ± 0.007	0.893 ± 0.010
Monks1	0.696 ± 0.004	0.776 ± 0.004	**0.781 ± 0.013**	0.780 ± 0.004	0.775 ± 0.001	0.753 ± 0.010	0.740 ± 0.013
Monks2	0.556 ± 0.032	0.796 ± 0.031	**0.802 ± 0.021**	0.789 ± 0.037	0.793 ± 0.034	0.770 ± 0.010	0.749 ± 0.008
Tic-tac-toe	0.568 ± 0.003	0.581 ± 0.006	0.576 ± 0.007	0.580 ± 0.008	0.586 ± 0.006	**0.587 ± 0.011**	0.563 ± 0.009
Krkopt	0.523 ± 0.004	0.546 ± 0.023	0.546 ± 0.012	0.531 ± 0.019	0.529 ± 0.003	0.537 ± 0.010	**0.549 ± 0.007**
Adult	0.791 ± 0.010	0.852 ± 0.020	**0.861 ± 0.008**	0.832 ± 0.007	0.828 ± 0.005	0.849 ± 0.009	0.837 ± 0.007
Connect4	0.634 ± 0.009	**0.681 ± 0.012**	0.676 ± 0.011	0.663 ± 0.010	0.659 ± 0.014	0.656 ± 0.009	0.673 ± 0.019
Census	0.751 ± 0.013	0.810 ± 0.007	**0.816 ± 0.013**	0.791 ± 0.016	0.742 ± 0.024	0.731 ± 0.006	0.748 ± 0.003
Zoo	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**
DNAPromoter	0.949 ± 0.014	**0.973 ± 0.016**	0.969 ± 0.011	0.931 ± 0.015	0.934 ± 0.013	0.956 ± 0.022	0.946 ± 0.012
Lymphography	0.861 ± 0.011	0.882 ± 0.010	**0.885 ± 0.010**	0.867 ± 0.007	0.873 ± 0.012	0.871 ± 0.009	0.869 ± 0.006
Audiology	0.742 ± 0.017	0.773 ± 0.018	**0.784 ± 0.015**	0.739 ± 0.017	0.759 ± 0.011	0.769 ± 0.009	0.736 ± 0.015
Housevotes	0.945 ± 0.010	0.975 ± 0.006	**0.983 ± 0.005**	0.952 ± 0.005	0.962 ± 0.017	0.977 ± 0.016	0.978 ± 0.009
Spect	0.603 ± 0.011	0.642 ± 0.014	**0.659 ± 0.029**	0.598 ± 0.011	0.630 ± 0.003	0.626 ± 0.018	0.637 ± 0.014
Soybeanlarge	0.803 ± 0.005	0.846 ± 0.014	**0.853 ± 0.014**	0.827 ± 0.009	0.836 ± 0.011	0.818 ± 0.008	0.822 ± 0.011
DNANominal	0.921 ± 0.008	**0.953 ± 0.017**	0.949 ± 0.009	0.932 ± 0.005	0.941 ± 0.017	0.924 ± 0.008	0.935 ± 0.017
Splice	0.860 ± 0.010	**0.898 ± 0.012**	0.894 ± 0.010	0.885 ± 0.011	0.880 ± 0.009	0.868 ± 0.004	0.877 ± 0.013
Krvskp	0.950 ± 0.013	0.975 ± 0.001	**0.976 ± 0.002**	0.959 ± 0.005	0.967 ± 0.003	0.924 ± 0.013	0.954 ± 0.003
Led24	0.642 ± 0.007	**0.713 ± 0.005**	0.689 ± 0.018	0.676 ± 0.013	0.662 ± 0.012	0.678 ± 0.014	0.655 ± 0.011

TABLE VI. DIFFERENT MODELS' PERFORMANCE ON **classification** ACCURACY (MEAN ± STANDARD DEVIATION)

Dataset	Euclidean	CPML-single	CPML-multi	KDML	LMNN	HELIC	DM3
Car	0.969 ± 0.013	0.970 ± 0.012	**0.977 ± 0.010**	**0.977 ± 0.012**	0.975 ± 0.003	0.969 ± 0.005	0.966 ± 0.004
Balance	0.866 ± 0.031	0.938 ± 0.030	0.929 ± 0.023	0.936 ± 0.031	0.940 ± 0.005	**0.941 ± 0.006**	0.937 ± 0.007
Mushroom	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**
Voting	0.938 ± 0.032	0.943 ± 0.043	0.935 ± 0.037	0.915 ± 0.112	0.923 ± 0.007	0.939 ± 0.007	0.924 ± 0.007
Nursery	0.983 ± 0.003	0.988 ± 0.003	**0.994 ± 0.003**	0.989 ± 0.003	0.985 ± 0.010	0.993 ± 0.008	0.984 ± 0.006
Monks1	0.842 ± 0.006	0.849 ± 0.005	**0.859 ± 0.010**	0.856 ± 0.007	0.853 ± 0.003	0.839 ± 0.017	0.845 ± 0.009
Monks2	0.711 ± 0.001	0.721 ± 0.002	**0.723 ± 0.002**	0.719 ± 0.006	0.717 ± 0.003	0.709 ± 0.005	0.713 ± 0.005
Tic-tac-toe	0.862 ± 0.006	0.874 ± 0.001	0.860 ± 0.007	0.880 ± 0.010	0.876 ± 0.008	**0.883 ± 0.007**	0.864 ± 0.010
Krkopt	0.513 ± 0.012	0.519 ± 0.019	0.523 ± 0.007	0.508 ± 0.017	0.514 ± 0.005	**0.548 ± 0.007**	0.537 ± 0.009
Adult	0.837 ± 0.011	0.853 ± 0.010	**0.854 ± 0.010**	0.839 ± 0.011	0.836 ± 0.006	0.845 ± 0.007	0.823 ± 0.008
Connect4	0.561 ± 0.007	**0.571 ± 0.016**	0.569 ± 0.009	0.564 ± 0.007	0.568 ± 0.012	0.546 ± 0.007	0.556 ± 0.023
Census	0.671 ± 0.007	**0.687 ± 0.009**	0.685 ± 0.009	0.681 ± 0.012	0.675 ± 0.009	0.667 ± 0.007	0.677 ± 0.016
Zoo	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**
DNAPromoter	0.896 ± 0.010	**0.919 ± 0.007**	0.916 ± 0.006	0.903 ± 0.008	0.902 ± 0.010	0.907 ± 0.009	0.904 ± 0.012
Lymphography	0.857 ± 0.005	**0.879 ± 0.008**	0.875 ± 0.010	0.865 ± 0.005	0.871 ± 0.009	0.866 ± 0.009	0.855 ± 0.003
Audiology	0.682 ± 0.009	0.701 ± 0.009	**0.713 ± 0.009**	0.671 ± 0.011	0.647 ± 0.009	0.708 ± 0.009	0.710 ± 0.018
Housevotes	0.906 ± 0.015	**0.941 ± 0.011**	0.932 ± 0.018	0.925 ± 0.017	0.921 ± 0.020	0.922 ± 0.008	0.931 ± 0.014
Spect	0.531 ± 0.007	0.549 ± 0.004	0.552 ± 0.009	0.538 ± 0.015	0.537 ± 0.005	0.542 ± 0.005	**0.558 ± 0.013**
Soybeanlarge	0.859 ± 0.018	0.879 ± 0.009	**0.885 ± 0.004**	0.869 ± 0.007	0.871 ± 0.009	0.869 ± 0.006	0.870 ± 0.015
DNANominal	0.871 ± 0.032	0.928 ± 0.021	**0.936 ± 0.018**	0.908 ± 0.015	0.924 ± 0.007	0.919 ± 0.007	0.909 ± 0.007
Splice	0.834 ± 0.009	0.879 ± 0.007	**0.881 ± 0.012**	0.851 ± 0.009	0.862 ± 0.007	0.868 ± 0.006	0.846 ± 0.022
Krvskp	0.930 ± 0.010	0.945 ± 0.003	**0.946 ± 0.008**	0.918 ± 0.035	0.924 ± 0.009	0.923 ± 0.007	0.915 ± 0.011
Led24	0.597 ± 0.019	0.642 ± 0.007	**0.651 ± 0.031**	0.601 ± 0.008	0.621 ± 0.019	0.648 ± 0.008	0.643 ± 0.007

[24] S. Q. Le and T. B. Ho, "An association-based dissimilarity measure for categorical data," *Pattern Recognition Letters*, vol. 26, no. 16, pp. 2549–2557, 2005.

[25] D. Ienco, R. G. Pensa, and R. Meo, "Context-based distance learning for categorical data clustering," in *International Symposium on Intelligent Data Analysis*. Springer, 2009, pp. 83–94.

[26] ——, "From context to distance: Learning dissimilarity for categorical data clustering," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, p. 1, 2012.

[27] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

APPENDIX A
PROOF OF PROPERTY 1

For the categorical projected multi distance (CPm), we have

$$
\begin{aligned}
&d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
&= \sum_{c=1}^{C} \sum_{p,q} A_{c,pq}^{ij} M_{c,pq} \\
&= \sum_{c=1}^{C} (\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))^{\top} M_c (\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j)) \geq 0 \quad (10)
\end{aligned}
$$

For the triangle-inequality, we first docompose the positive semi-definite matrix as $M_c = Q^T \Lambda Q$, where Q is the orthogonal matrix and Λ is the diagonal matrix as $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_D)$ and $\lambda_1 \geq 0, \ldots, \lambda_D \geq 0$. Further, we let matrix U to be defined as $U = \mathrm{diag}(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_D}) Q$. Thus,

we have $M_c = U^\top U$. Let $\widehat{\phi}(\boldsymbol{x}_i) = U\phi(\boldsymbol{x}_i)$, it is easily to see that

$$
\begin{aligned}
&(\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))^\top M_c(\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j)) \\
=&(\widehat{\phi}_c(\boldsymbol{x}_i) - \widehat{\phi}_c(\boldsymbol{x}_j))^\top (\widehat{\phi}_c(\boldsymbol{x}_i) - \widehat{\phi}_c(\boldsymbol{x}_j)) \\
=&\sum_d (\widehat{\phi}_{cd}(\boldsymbol{x}_i) - \widehat{\phi}_{cd}(\boldsymbol{x}_j))^2
\end{aligned}
\tag{11}
$$

where $\widehat{\phi}_{cd}(\boldsymbol{x}_i)$ refers to the d-th element of $\widehat{\phi}_c(\boldsymbol{x}_i)$. Eq. (11) shows the distance can be alternatively represented as a Euclidean distance format. Based on the triangle-inequality of the Euclidean distance, we can straightforwardly get the triangle-inequality of the categorical projected multi-distance.

For the categorical projected single distance (CPs), we have

$$
\begin{aligned}
&d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
=&C\sum_{p,q} A^{ij}_{pq} M_{pq} \\
=&\sum_{c=1}^{C} (\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j))^\top M(\phi_c(\boldsymbol{x}_i) - \phi_c(\boldsymbol{x}_j)) \geq 0
\end{aligned}
\tag{12}
$$

The triangle inequality of the categorical projected single distance (CPs) can be obtained in a similar way.