On the Benefits of Two Dimensional Metric Learning

Di Wu*, Fan Zhou*, Boyu Wang, Qicheng Lao, Chi Man Wong, Changjian Shui, Yuan Zhou, and Feng Wan

Abstract—In this paper, we study two dimensional metric learning (2DML) for matrix data from both theoretical and algorithmic perspectives. We first investigate the generalization bounds of 2DML based on the notion of Rademacher complexity, which theoretically justifies the benefits of learning from matrices directly. Furthermore, we present a novel boosting-based algorithm that scales well with the feature dimension. Finally, we introduce an efficient rank-one correction algorithm, which is tailored to our boosting learning procedure to produce a low-rank solution to 2DML. As our algorithm works directly on the data in matrix representation, it scales well with the feature dimension, keeps the structure and dependence in the data, and has a more compact structure and much fewer parameters to optimize. Extensive evaluations on several benchmark data sets also empirically verify the effectiveness and efficiency of our algorithm.

Index Terms—Two dimensional learning, metric learning, Rademacher complexity, boosting, low-rank matrices.

1 Introduction

Distance metric learning plays an important role in many machine learning algorithms since their performances rely on distance metrics for an appropriate definition of similarity/dissimilarity over the input space (e.g., k-nearest neighbor (k-NN) classifier). Starting from [44], distance metric learning has been actively studied as a promising approach to learn a problem-specific distance metric [11], [36], [41]. In the usual setting, the data is provided as a collection of vectors $\{x_i\}_{i=1}^N \in \mathbb{R}^d$, and the objective of metric learning is to learn a Mahalanobis distance parameterized by a *symmetric positive definite* (SPD) matrix $M \in \mathbb{S}_+^d$: $d_M(x_i, x_j) = \sqrt{(x_i - x_j)^\top M (x_i - x_j)}$, where $\mathbb{S}_+^d$ denotes the cone of SPD matrices defined over $\mathbb{R}^{d\times d}$, such that additional constraints (e.g., pairwise similarity, structural constraints) are satisfied. However, there are many real world applications, such as image classification [45], electroencephalogram (EEG) signal analysis [40], auditory spectrogram analysis [25], [29], where the instances are formed as matrices. When dealing with these problems, classical metric learning algorithms have to *vectorize* the matrices, which has three fundamental limitations:

1. The vectorized features are usually high dimensional, which makes the learning algorithms infeasible due to the time and space complexity. For example, vectorizing a $d \times d$ matrix results in a d^2 vector, which requires $\mathcal{O}(d^6)$ at each optimization step [41].
2. The model complexity (i.e., the SPD matrix M) is of size $\mathcal{O}(d^4)$, which indicates that the classical algorithms have higher risk of overfitting.
3. Even though the high dimensionality problem can be alleviated by some preprocessing step such as principal component analysis (PCA), the natural structure of the data is lost due to vectorization. For example, when analyzing multichannel EEG signals of size $m \times n$, where m is the number of channels, and n is the number of time points, vectorizing the data will cause the loss of either temporal or spatial information, which could be potentially useful for the analysis.

Aiming to solve these problems, there have been extensive studies on *two dimensional* (sometimes also termed as *bilinear*) learning algorithms in the literature, which can deal with matrix data directly. This idea was first proposed in [45], which was a two dimensional extension of classical PCA. Later, it was adopted in discriminant analysis [8], [10], [46], canonical correlation analysis [21], support vector machines [16], [32], logistic regression [39], local tensor discriminant analysis [27], and metric learning [38]. Recently, bilinear learning has also studied in deep learning [47]. In practice, these techniques have been applied to feature extraction and dimensionality reduction [28], [48], image

*Equal contribution. The authors are listed in alphabetical order. (Corresponding author: B. Wang.)

- *D. Wu is with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada. E-mail: di.wu5@mail.mcgill.ca.*
- *F. Zhou is with the Department of Computer Science, Université Laval, Quebec City, QC, Canada. E-mail: fan.zhou.1@ulaval.ca.*
- *B. Wang is with the Department of Computer Science and the Brain Mind Institute, University of Western Ontario, London, ON, Canada, and also with the Vector Institute, Toronto, ON, Canada. E-mail: bwang@csd.uwo.ca.*
- *Q. Lao is with the West China Biomedical Big Data Center, West China Hospital of Sichuan University, Chengdu, China. E-mail: qicheng.lao@gmail.com.*
- *C. Shui is with the Department of Electrical and Computer Engineering, Université Laval, Quebec City, QC, Canada. E-mail: changjian.shui.1@ulaval.ca.*
- *Y. Zhou is with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China. E-mail: zhouyuan@tju.edu.cn.*
- *C. M. Wong and F. Wan are with the Department of Electrical and Computer Engineering, University of Macau, Macau SAR, China and also with the Centre for Cognitive and Brain Sciences, Institute of Collaborative Innovation, University of Macau, Macau SAR, China, and also with the Centre for Artificial Intelligence and Robotics, Institute of Collaborative Innovation, University of Macau, Taipa, Macau SAR, China, E-mail: chiman465@gmail.com, fwan@um.edu.mo.*

classification [15], [46], EEG signal analysis [8], fMRI signal analysis [1], and music genre classification [29].

While the empirical results presented in these works have verified the effectiveness of the two dimensional learning approach, its theoretical justification, to the best of our knowledge, is *never* investigated. As the main contribution, we theoretically analyze the benefits of *two dimensional metric learning* (2DML), showing that the Rademacher complexity of 2DML has a faster convergence rate than traditional metric learning algorithms when dealing with matrix data. Then, we present an efficient 2DML algorithm to further alleviate the computational issue of metric learning, which leads to our second contribution. More specifically, the proposed algorithm alternatingly learns two SPD matrices, one for each dimension of the matrix data. To learn each metric matrix, we adopt the idea that a SPD matrix can be formulated as a linear positive combination of trace-one rank-one (TORO) matrices [36], and can be solved efficiently by using an AdaBoost-like procedure. In addition, we also introduce an efficient rank-one correction algorithm to learning a low-rank metric matrix, which is particularly useful for reducing the model complexity.

We note that two-dimensional data may also be handled by advanced deep learning techniques, such as convolutional neural networks [13]. However, there are still many small size problems in practice, which cannot be properly addressed by deep nets. For deep metric learning, if the number of instances of each class is too small, it will be challenging to form training batches to build a reliable model. Therefore, the primary focus of this work is linear metric learning, but our analysis can also be generalized to deep metric learning [18] by leveraging the recent theoretical results of deep nets in [12].

2 Two Dimensional Metric Learning

For a better presentation of our theory and algorithm, Table 1 summarizes the notations used in this paper.

2.1 Problem Setup

In the one-dimensional metric learning setting, let $S = \{z_i = (x_i, y_i)\}_{i=1}^N$ be a data set of size N drawn from some distribution $\mathcal{D}$, where $x_i \in \mathbb{R}^d$ is a feature vector, and $y_i \in \{1, \ldots, C\}$ is its class label. Let $\mathcal{H}$ be a hypothesis class of metric functions, then, the goal of metric learning is to minimize the following objective function:

$$\min_{h \in \mathcal{H}} \mathcal{L}_S(h),$$

where $\mathcal{L}_S(h) = \frac{1}{N(N-1)} \sum_{i \neq j}^N \ell(h, z_i, z_j)$ is empirical loss, $\ell(\cdot)$ is the loss function of h over the pairs of examples (e.g., hinge loss [41], [42]).[1] In one-dimensional metric learning, h can be parameterized by the Mahalanobis distance between x_i and x_j:

$$h(x_i, x_j) = (x_i - x_j)^\top M (x_i - x_j) = ||L(x_i - x_j)||_2^2, \quad (1)$$

where $M = L^\top L \in \mathbb{S}_+^d$ is a SPD matrix. Eq. (1) indicates that learning M is equivalent to learning a linear transformation parameterized by $L \in \mathbb{R}^{d' \times d}$: $\hat{x} = Lx$, so that $\{(\hat{x}_i, y_i)\}_{i=1}^N$ satisfies the constraints in the transformed space.

In the setting of 2DML, we are given $S = \{(X_i, y_i)\}_{i=1}^N$, where $X_i \in \mathbb{R}^{m \times n}$ is an instance in a matrix form. For example, in EEG signal analysis, the data can be represented as multichannel time series, where m is the number of channels, and n is the number of sampling points. Accordingly, we aim to learn a bilinear transformation parameterized by $L \in \mathbb{R}^{m' \times m}$ and $R \in \mathbb{R}^{n' \times n}$: $h(X_i, X_j) = ||L(X_i - X_j)R^\top||_{\mathsf{F}}^2$, where $||\cdot||_{\mathsf{F}}$ is the Frobenius norm of a matrix. By the property of Frobenius norm, we note that

$$\begin{aligned} ||L(X_i - X_j)R^\top||_{\mathsf{F}}^2 &= \left\langle (X_i - X_j)^\top U (X_i - X_j), V \right\rangle \quad (2) \\ &= \left\langle (X_i - X_j) V (X_i - X_j)^\top, U \right\rangle, \end{aligned}$$

where $U = L^\top L \in \mathbb{R}^{m \times m}$ and $V = R^\top R \in \mathbb{R}^{n \times n}$, $\langle A, B \rangle = \mathrm{Tr}(A^\top B)$, and $\mathrm{Tr}(\cdot)$ is the trace of a matrix. To control the model complexity, we further require $||U||_{\mathsf{F}} \leq \alpha$, and $||V||_{\mathsf{F}} \leq \beta$. Consequently, the optimization problem of 2DML becomes

$$\min_{||U||_{\mathsf{F}} \leq \alpha, ||V||_{\mathsf{F}} \leq \beta} \mathcal{L}_S(U, V). \quad (3)$$

2.2 Theory

In this section, we analyze the generalization performance of 2DML by exploiting the notion of Rademacher complexity [2]. In particular, we show that the Rademacher complexity of a two-dimensional metric algorithm is smaller than that of one-dimensional approach, which justifies the benefits of 2DML.

TABLE 1
Summary of Notations

Notation	Description
X	An $m \times n$ dimensional instance in a matrix form
y	The label of an instance
z	A pair of an instance and its label
S	A data set of size N
$\mathcal{D}$	The data distribution where S is sampled from
N	The number of instances
$\lfloor \frac{N}{2} \rfloor$	The largest integer less than or equal to $\frac{N}{2}$
$\mathcal{C}$	The set of constraints
$\mathbb{S}_+^d$	The cone of $d \times d$ SPD matrices
$\grave{}$, ^{1}	The set of trace-one rank-one matrices
	A loss function
	A Lipschitz constant
$\mathcal{H}$	A hypothesis class of metric functions
h	A metric function in the hypothesis class $\mathcal{H}$: $h \in \mathcal{H}$
$\mathcal{G}$	The hypothesis class of composition of $h \in \mathcal{H}$ and $\grave{}$
$\mathcal{L}_S$	The empirical loss over the data set S
$\mathcal{L}_D$	The expected loss over the distribution $\mathcal{D}$
$L; R$	Left and right projection matrices
$U; V$	$U = L^\top L; V = R^\top R$: left and right metric matrices
$;$	The upper bounds of the Frobenius norm of U and V
	The upper bound of the Frobenius norm of two instances
	The upper bound of the spectral norm of two instances
T	Maximum number of iterations for TDBML (Algorithm 1)
K	Maximum number of iterations for BML (Algorithm 2)
$_U$; $_V$	regularisation parameters for TDBML and BML
	The desired rank of U or V (Algorithm 3)
Q	Current rank of U or V
$i; j; l$	Instance indices
c	Constraint index
t	Iteration index for TDBML
k	Iteration index for DBML
q	TORO matrix index

1. For simplicity, our theoretical analysis focus on pair-based constraints, but the conclusion is also applicable to triplet-based metric learning algorithms with slightly modification of the proof schema.

In this paper, we assume that for any pair of instances X, X', we have $||X - X'||_{\mathsf{F}} \leq \eta$ and $||X - X'||_2 \leq \mu$, where $||\cdot||_{\mathsf{F}}$ and $||\cdot||_2$ are, respectively, the Frobenius norm and the spectral norm of a matrix. In addition, we consider ρ-Lipschitz continuous loss functions, as defined below.

Definition 1 (ρ**-Lipschitz continuity).** *Let $\mathcal{H}$ be a hypothesis class defined over pairs of examples. A loss function $\ell(h, z, z')$ is ρ-Lipschitz continuous with respect to $\mathcal{H}$ for some $\rho \in \mathbb{R}_+$ if, for any two hypotheses $h, h' \in \mathcal{H}$ and for any $z, z' \sim \mathcal{D}$, we have:*

$$|\ell(h, z, z') - \ell(h', z, z')| \leq \rho\, |h(z, z') - h'(z, z')|\,.$$

The main obstacle to derive generalization bound for metric learning is the fact that the training examples for metric learning algorithms are formed by the pairs of the instances, and therefore cannot be independent. Consequently, if one instance is changed, some other pairs/triplets are also changed. To address this issue, we introduce the notion of Rademacher complexity defined over sums of independent and identically distributed (i.i.d.) sample blocks.

Definition 2 (**Empirical Rademacher complexity).** *Let $\mathcal{G}$ be a hypothesis class defined over pairs of examples, and $S = \{z_1, \ldots, z_N\}$ be a fixed sample of size N. Then, the* empirical Rademacher complexity *of $\mathcal{G}$ with respect to the sample S is defined as:*

$$\hat{\mathfrak{R}}_S(\mathcal{G}) = \mathbb{E}_\sigma \left[\sup_{g \in \mathcal{G}} \frac{1}{\lfloor \frac{N}{2} \rfloor} \sum_{i=1}^{\lfloor \frac{N}{2} \rfloor} \sigma_i g(z_i, z_{\lfloor \frac{N}{2} \rfloor + i}) \right],$$

where $\sigma_i, \ldots, \sigma_{\lfloor \frac{N}{2} \rfloor}$ are independent uniform random variables taking values in $\{-1, +1\}$, and $\lfloor \frac{N}{2} \rfloor$ is the largest integer less than or equal to $\frac{N}{2}$.

Remark 1. *Our definition of Rademacher complexity is related to the Rademacher average for metric learning in [7]. It is worth mentioning that the Rademacher average is only defined over examples, whereas Definition 3 takes the hypothesis class $\mathcal{G}$ into account, which allows us to capture and analyze the richness of $\mathcal{G}$ for metric learning.*

Definition 3 (**Rademacher complexity).** *Let $\mathcal{D}$ be the distribution according to which samples are drawn. Then, the* Rademacher complexity *is the expectation of the empirical Rademacher complexity $\hat{\mathfrak{R}}_S(\mathcal{G})$ over the sample S drawn according to $\mathcal{D}$:*

$$\mathfrak{R}(\mathcal{G}) = \mathbb{E}_{S \sim \mathcal{D}^N} \hat{\mathfrak{R}}_S(\mathcal{G}).$$

Given the definitions, the following theorem shows that the generalization performance of metric learning depends on $\mathfrak{R}(\mathcal{G})$.[2]

Theorem 1. *Let $\mathcal{H}$ be a hypothesis class of metric functions, ℓ be a ρ-Lipschitz continuous loss function bounded by $B \geq 0$, $S = \{z_1, \ldots, z_N\}$ be a fixed sample of size N drawn from a distribution $\mathcal{D}$, and $\mathcal{L}_{\mathcal{D}}(h) = \mathbb{E}_{z, z' \sim \mathcal{D}}[\ell(h, z, z')]$ be the expected loss of h over $\mathcal{D}$. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the followings hold for any $h \in \mathcal{H}$:*

$$\mathcal{L}_{\mathcal{D}}(h) \leq \mathcal{L}_S(h) + 2\rho\mathfrak{R}(\mathcal{H}) + 2B\sqrt{\frac{\log \frac{1}{\delta}}{2N}}, \tag{4}$$

2. All proofs are delegated to the supplementary materials.

Theorem 1 shows that if the Rademacher complexity of $\mathcal{H}$ is of order $O(\frac{1}{\sqrt{N}})$, as we will prove in Theorem 2, metric learning will have a convergence rate of order $O(\frac{1}{\sqrt{N}})$, which is the same as traditional supervised learning. Moreover, the generalization bounds above are model-agnostic – they are applicable to both traditional metric learning and 2DML.

To derive meaningful generalization bounds and reveal the benefits of 2DML, we upper bound the Rademacher complexity of 2DML, which leads to the main theoretical contribution of this work.

Theorem 2. *Let $S = \{z_1, \ldots, z_N\}$ be a sample of size N, and $\mathcal{H}$ be the hypothesis class parameterized by $U \in \mathbb{S}_+^m$ and $V \in \mathbb{S}_+^n$ as in (2), with $||U||_{\mathsf{F}} \leq \alpha$ and $||V||_{\mathsf{F}} \leq \beta$. Then, the Rademacher complexity of $\mathcal{H}$ can be upper bounded by*

$$\mathfrak{R}_S(\mathcal{H}) \leq \frac{\alpha\beta\eta^2}{\sqrt{\lfloor \frac{N}{2} \rfloor}}. \tag{5}$$

and

$$\mathfrak{R}_S(\mathcal{H}) \leq \alpha\beta\mu^2 \sqrt{\frac{2\log(m^2 + n^2)}{\lfloor \frac{N}{2} \rfloor}}. \tag{6}$$

If $m = n$, then, by the definition of the Frobenius and spectral norms, we have $\mu \leq \eta \leq \sqrt{n}\mu$. Consequently, the ratio of the bounds in (5) and (6) is between $O(\frac{1}{\log n})$ and $O(\frac{n}{\log n})$. In other words, the bound in (6) could be much tighter than in (5) especially for high dimensional data. Additionally, the benefits of learning from matrices directly are more apparent in (6) – when dealing with vectorized data, the bound in (6) becomes $\sqrt{2\log(m^2 n^2 + 1)/\lfloor \frac{N}{2} \rfloor}$, which implies a slower convergence rate than 2DML.

The theoretical properties of metric learning have been investigated in the literature. In particular, the generalization bounds have been studied by making use of the notions of algorithmic stability [17], [30], Rademacher complexity [7], [14], [22], good similarities [4], algorithmic robustness [3]. In fact, if we consider one-dimensional metric learning by letting $n = 1$ and $\beta = 1$, we will recover the Rademacher bounds for metric learning in [7], [22] as a special case of our analysis. Compared to the Rademacher bounds using U-statistics in [7], [14], [22], which are only applicable to hinge loss or strongly convex function, our results are more general and can be applied to any Lipschitz continuous loss function. Most importantly, the Rademacher average in [7], [14] is only defined over examples, while the Rademacher complexity defined in our work is also associated with a hypothesis space, which enables us to analyze 2DML. To the best of our knowledge, none of existing works has studied the benefits of 2DML from a theoretical perspective.

2.3 Algorithm

In this section, we present an efficient and scalable algorithm, termed *two dimensional boosted metric learning* (TDBML) to solve (3). In particular, we consider the relative constraint defined over triplets as in [37], [41]. In particular, we define

$$X_c^U = (X_i - X_l)^\top U (X_i - X_l) - (X_i - X_j)^\top U (X_i - X_j); \tag{7}$$

Algorithm 1 TDBML

Input: Data set S, λ_U; λ_V: regularization parameters, T: maximum number of iterations, K: maximum number of boosting iterations for BML.

Initialize $V^{(0)}; U^{(0)}$
for $t = 1; \ldots; T$ **do**
  Update X_c^V given $V^{(t-1)}$ using Eq. (8).
  Call $U^{(t)} = \text{BML}(X_c^V; \lambda_U; K; U^{(t-1)})$ to solve Eq. (11).
  Update X_c^U given $U^{(t)}$ using Eq. (7).
  Call $V^{(t)} = \text{BML}(X_c^U; \lambda_V; K; V^{(t-1)})$ to solve Eq. (23).
  if converge **then** break **end if**
end for

Output: Metric matrices $U = U^{(t)}$ and $V = V^{(t)}$.

Algorithm 2 $\text{BML}(X_c, \lambda, K, M_0)$

Input: Triplet set $\{X_c\}_{c=1}^{|\mathcal{C}|}$, λ: regularization parameter, K: maximum number of iterations, M_0: an initial matrix.

Initialize ξ_c using Eq. (21), where $\xi_c = \langle X_c; M_0 \rangle$.
for $k = 1; \ldots; K$ **do**
  Compute γ_k using Eq (18).
  Select a new base learner P_k using Eq (19).
  if $(\gamma_k) < 0$ **then** break **end if**
  Compute the coefficients w_k by solving Eq (20).
  Call ROC to learn a low-rank metric (optional).
  Update the weight π_c for each triplet X_c using (22).
end for

Output: Metric matrix $M = M_0 + \sum_{k=1}^{K} w_k P_k$.

and

$$X_c^V = (X_i - X_l)V(X_i - X_l)^\top - (X_i - X_j)V(X_i - X_j)^\top; \quad (8)$$

where $c = 1, \ldots |\mathcal{C}|$, $\mathcal{C}$ is the set of constraints formed by the labels of instances, and $|\mathcal{C}|$ is the number of the constraints or any other side-information. For example, X_j can be an instance having the same label as X_i, and X_l can be an instance having a different label with X_i. According to (2), the relative constraint we aim to optimize becomes

$$\phi_c = \begin{cases} \phi_c^U, & \langle X_c^U, V \rangle \\ \phi_c^V, & \langle X_c^V, U \rangle \end{cases}. \quad (9)$$

The objective of TDBML is to learn U and V such that the margin ϕ_c is as large as possible. Inspired by [36], [37], we impose the logarithm of the sum of the exponential (log-sum-exp) loss to derive our boosting-based algorithm. To this end, the objective function of TDBML can be formulated as

$$\min_{U,V} \log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c\right) + \frac{\lambda_U}{2}||U||_{\mathsf{F}}^2 + \frac{\lambda_V}{2}||V||_{\mathsf{F}}^2 \quad (10)$$
$$\text{s.t. } U, V \in \mathbb{S}_+^d,$$

where instead of explicitly controlling the Frobenius norm of U and V, we have introduced additional regularization terms, and λ_U and λ_V are the regularization parameters.

As the problem (10) is a bilinear optimization problem, it is difficult to optimize U and V simultaneously. Instead, we adopt the alternating optimization as in [38], [46], leading to an iterative algorithm in the following. Note that if U (V) is fixed, solving (10) with respect to V (U) is a convex optimization problem, and can be solved by using the similar procedure as in [37].

2.3.1 Computing U given V

Given V, the objective function of TDBML becomes

$$\min_{U} \log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c^V\right) + \frac{\lambda_U}{2}||U||_{\mathsf{F}}^2 \quad (11)$$
$$\text{s.t. } U \in \mathbb{S}_+^d.$$

As a positive semidefinite matrix can be decomposed by a positive linear combination of TORO matrices, we can reformulate U as $U = \sum_{k=1}^{K} w_k P_k$, with $w_k \geq 0$, $\text{rank}(P_k) = 1$, and $\text{Tr}(P_k) = 1, \forall k = 1, \ldots, K$. Then, we have

$$\phi_c^V = \sum_{k=1}^{K} w_k \left\langle X_c^V, P_k \right\rangle = \xi_c^\top w, \quad (12)$$

where $w = [w_1; \ldots; w_K]^\top$, $\xi_c = [\langle X_c^V; P_1 \rangle; \ldots; \langle X_c^V; P_K \rangle]^\top$. Now the objective function (11) becomes

$$\min_{w,\phi} \log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c^V\right) + \frac{\lambda_U}{2}\left|\left|\sum_{k=1}^{K} w_k P_k\right|\right|_{\mathsf{F}}^2 \quad (13)$$
$$\text{s.t. } \phi_c^V = \xi_c^\top w, \; P_k \in \Omega_1, w \succeq 0,$$

where Ω_1 is the set of TORO matrices. The Lagrange function of (13) is

$$L(w, \phi, \mu, \nu) = \log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c^V\right) + \frac{\lambda_U}{2}\left|\left|\sum_{k=1}^{K} w_k P_k\right|\right|_{\mathsf{F}}^2 + \sum_{c=1}^{|\mathcal{C}|} \mu_c \left(\phi_c^V - \xi_c{}^\top w\right) - \nu^\top w, \quad (14)$$

where $\mu_c \geq 0, c = 1, \ldots, |\mathcal{C}|, \nu \succeq 0$ are the sets of Lagrange multipliers. Then, the dual function is given by

$$\inf_{w,\phi} L = \inf_{\phi}\left[\log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c^V\right) + \sum_{c=1}^{|\mathcal{C}|} \mu_c \phi_c^V\right] + \inf_{w}\left[\frac{\lambda_U}{2}\left|\left|\sum_{k=1}^{K} w_k P_k\right|\right|_{\mathsf{F}}^2 - \left(\sum_{c=1}^{|\mathcal{C}|} \mu_c \xi_c^\top + \nu^\top\right) w\right]. \quad (15)$$

It can be observed that the problem (15) can be minimized with respect to ϕ and w separately, which gives the Lagrange dual problem of (11)

$$\max_{\mu} -\sum_{c=1}^{|\mathcal{C}|} \mu_c \log \mu_c \quad (16)$$
$$\text{s.t.} \quad \sum_{c=1}^{|\mathcal{C}|} \mu_c = 1, \;\; \mu_c \geq 0, \forall c = 1, \ldots, |\mathcal{C}| \quad \text{(C1)}$$
$$\sum_{c=1}^{|\mathcal{C}|} \mu_c \xi_{c,k} \leq \lambda_U \sum_{i \neq k} \langle w_i P_i, P_k \rangle, \quad \text{(C2)}$$

where $\xi_{c;k} = \langle X_c^V; P_k \rangle$ is the k-th element of ξ_k. However, as the number of possible TORO matrices is infinite, either

the primal (11) or dual (16) can be solved. Following the intuition of *column generation* [24], we use an AdaBoost-like algorithm which sequentially adds TORO matrices to the current solution. More specifically, at the k-th iteration, a TORO matrix that most violates the constraints (C2) is added to the optimization problem. In other words, we have to solve

$$P_k = \arg\max_{P} \left\langle \frac{1}{\lambda_U} \sum_{c=1}^{|\mathcal{C}|} \mu_c X_c^V - H_{k-1}, P \right\rangle, \quad \text{s.t. } P \in \Omega_1, \tag{17}$$

where $H_k = \sum_{i=1}^{k} w_i P_i$ is the solution of H at the k-th boosting iteration. Note that as P_k is a TORO matrix, the solution of (17) can be obtained by eigenvalue decomposition. More specifically, let

$$\Sigma_k = \frac{1}{\lambda_U} \sum_{c=1}^{|\mathcal{C}|} \mu_c X_c^V - H_{k-1}, \tag{18}$$

and p_k is the eigenvector corresponding to its largest eigenvalue $\sigma(\Sigma_k)$. Then we have

$$P_k = p_k p_k^\top. \tag{19}$$

Once we add new basis to the optimization problem, its coefficient w_k can be found by minimizing the cost function of primal problem (11). By setting its derivative with respect to w_k to zero, we need to find w_k such that

$$\sum_{c=1}^{|\mathcal{C}|} \mu_{c,k-1} \theta_{c,k} \exp(-w_k \xi_{c,k}) = 0, \tag{20}$$

where $\theta_{c,k}$, $\xi_{c,k} - \lambda_U \langle P_k, H_{k-1} \rangle - \lambda_U w_k$, and $\mu_{c,k}$ is the updated value of μ_c at the k-th boosting iteration. Note that as the cost function is convex with respect to w_k, there exists a unique solution for (20), which can be found by any root-finding algorithm (e.g., bisection search [36]). In practice, we have found that the performance of the algorithm is not sensitive to the coefficient w_k, and therefore can also simply set to be a fixed small constant, as in ϵ-boosting [33].

It remains to show the update rule for $\mu_{c,k}$, which can be derived from the KKT conditions of (14). Setting the derivative of (14) with respect to ϕ_c^V to zero gives

$$\mu_c = \frac{\exp -\phi_c^V}{\sum_{c'=1}^{|\mathcal{C}|} \exp -\phi_{c'}^V}, \tag{21}$$

which suggests that once we add a new TORO matrix P_k as a weak learner to the current solution, $\mu_{c,k}$ is updated by

$$\mu_{c,k} = \frac{\exp\left(-\phi_{c,k-1}^V - w_k \langle X_c^V, P_k \rangle\right)}{Z_{k-1}} = \frac{\mu_{c,k-1} \exp\left(-w_k \langle X_c^V, P_k \rangle\right)}{Z'_{k-1}}, \tag{22}$$

where Z_{k-1} and Z'_{k-1} are a normalization factor such that $\sum_{c=1}^{|\mathcal{C}|} \mu_{c,k} = 1$. Note that the dual variable $\mu_{c,k}$ plays a role of the weight of each sample X_c^V at the k-th iteration as in AdaBoost, since each weak learner is obtained by computing the eigenvector corresponding to the largest eigenvalue of Σ_k, which is a linear combination of X_c^V, weighted by $\mu_{c,k}$, as shown in (17).

Note that the eigenvalue $\sigma(\Sigma_k)$ can also be used as the stop criterion of the algorithm, as $\sigma(\Sigma_k) < 0$ implying that no TORO matrix violates the constraint (C2), and hence the algorithm converges.

2.3.2 Computing V given U

Given U, the objective function of B2BML becomes

$$\min_V \log\left(\sum_{c=1}^{|\mathcal{C}|} \exp -\phi_c^U\right) + \frac{\lambda_V}{2} ||V||_{\mathsf{F}}^2 \quad \text{s.t. } V \in \mathbb{S}_+^d, \tag{23}$$

which can be solved by using the same optimization procedure as for (11), and therefore is omitted here.

The pseudo-code of TDBML is shown in Algorithm 1, which iteratively calls the subroutine *boosted metric learning* (BML) in Algorithm 2 to alternatingly update U and V.

2.4 Learning Low-Rank Metric

In practice, a low-rank solution of metrics is usually preferred since it can reduce the memory and model complexity, and therefore may avoid overfitting and improve the generalization performances. In addition, it also corresponds to low-rank projection matrices L and R, which map the data to a lower dimensional space with more compact representation. In TDBML, a low-rank solution can be obtained by limiting the maximum numbers of iterations T and K. However, the performances of this approach are usually very poor due to early stop without fully optimizing the objective function. To this end, we present *rank-one correction* (ROC), an efficient approach tailored specifically to BML, to explicitly controls the rank of a matrix. The pseudo-code of ROC is summarized in Algorithm 3. Specifically, let Υ be the desired rank of U or V. ROC maintains the current solution M_Q (M_Q can either be U or V) as a collection of Q TORO matrices: $M_Q = \sum_{q=1}^{Q} w_q P_q$. Once a new TORO matrix is added to M_Q and rank is larger than Υ, it evaluates the loss functions of all possible combinations of positive semi-definite matrices $M_{/q}$, where $M_{/q}$ is the metric without the q-th TORO matrix. Then, ROC keeps the $M_{/\tilde{q}}$ with the lowest loss and remove $P_{\tilde{q}}$ from the collection. Although ROC requires extra resources at each step, it makes TDBML converge within much fewer iterations since it terminates the algorithm if M_{Q+1} has the highest loss.

2.5 Computational Complexity

The main computation cost of the subroutine BML comes from the eigenvalue decomposition step (i.e., Eq (19)) which takes $\mathcal{O}(d^2)$ to find the largest eigenvalue and the corresponding eigenvector.[3] Updating X_c^U and X_c^V requires $\mathcal{O}(|\mathcal{C}|d^3)$ for full-rank matrices. If we set $K = 1$ for BML, only rank-one update is required, which takes $\mathcal{O}(|\mathcal{C}|d^2)$. If ROC is applied, it takes $\mathcal{O}(|\mathcal{C}|d^2\Upsilon)$ to evaluate the loss function. On the other hand, classical metric learning algorithms have to convert the matrices to the vectors, which makes them much more computational expensive for high-dimensional features. For example, it requires $\mathcal{O}(d^4)$ and

3. For the simplicity of analysis, we assume that $m = n = d$.

Algorithm 3 Rank-One Correction (ROC)

Input: Triplet set $\{\mathbf{X}_q\}_{q=1}^{|\mathcal{C}|}$, the desired rank r, the current metric matrix $M_Q = \sum_{q=1}^{Q} w_q P_q$, the new TORO matrix wP.

1: Let $w_{Q+1}P_{Q+1} = wP$.
2: **if** $\operatorname{rank}(M_Q) < r$ **then**
3: $M_Q = \sum_{q=1}^{Q+1} w_q P_q$, $Q \leftarrow Q+1$.
4: **else**
5: **for** $q = 1, \ldots, Q+1$ **do**
6: $M_{\neq q} = \sum_{q=1, i\neq q}^{Q+1} w_q P_q$
7: Compute $\texttt{loss}(M_{\neq q})$: the loss of $M_{\neq q}$ using Eq. (11) (or Eq. (23)).
8: **end for**
9: $q = \arg\min_q \texttt{loss}(M_{\neq q})$.
10: Replace $w_q P_q$ with wP.
11: **end if**

Output: The set of TORO matrices $\{w_q P_q\}_{q=1}^{Q}$, and the metric matrix $M_Q = \sum_{q=1}^{Q} w_q P_q$.

$\mathcal{O}(d^6)$ at each iteration for BoostMetric and LMNN respectively, as discussed in the section of introduction.

With respect to the memory usage, BoostMetric and LMNN requires $\mathcal{O}(|\mathcal{C}|d^4)$ to hold the outer product of the examples in each triplet, while TDBML only requires $\mathcal{O}(|\mathcal{C}|d^2)$. For instance, if we use a double-precision floating-point data type, it will take about 74.5 GB for BoostMetric to hold the outer product of 100 triplets of vectorized 100×100 images, while TDBML only needs 7.6 MB to store the data.

In terms of the convergence of our algorithms, BML follows BoostMetric, so it naturally inherits the convergence property of BoostMetric. TDBML is essentially an alternating optimization, so it can be trivially proved that TDBML will *monotonically* reduce the loss function, and hence it converges [5].

3 Experiments

In this section, we first evaluate TDBML on seven benchmark data sets of image classification, including COIL100, COIL 20, MNIST, ORL, USPS, extended Yale data sets, and the IMM data set. We also evaluate our methods on EEG content decoding tasks.

3.1 Implementation Details

While BML is mainly motivated by BoostMetric [36], [37], there are some major modifications to make the algorithm more efficient.

1) To make the algorithm converge, each time we call BML, we use the matrix learned at the previous step as starting point. In other words, the first base learner of the boosting algorithm is $U^{(t-1)}$ or $V^{(t-1)}$. By doing so, it can be trivially proved that TDBML *monotonically* reduces the loss function at each iteration.
2) While the boosting iterations K in [36], [37] can be very large (e.g., 1000), we set K so be a small number to make the algorithm more efficient. In the extreme case, we can simply set $K = 1$, resulting in a rank-one update at each iteration. In our experiments, we simply set $K = 10$, the number of iterations $T = 100$, and the regularization parameters $\lambda_U = \lambda_V = 10^{-4}$. The rank control parameter Υ is selected by cross-validation.
3) We use the same approach to generate the triplets as in [41]. After the metric learning step, the test examples are classified by using 1-nearest neighbour classifier. We run the experiments 10 times by randomly splitting the training/testing data set, and the average results are reported.

TABLE 2
Summary of the Data Sets used in Experiments

Data Set	#ex	#dim	#class
COIL100	7200	$32 \times 32 = 1024$	100
COIL20	1440	$32 \times 32 = 1024$	20
MNIST	4000	$28 \times 28 = 784$	10
ORL	400	$32 \times 32 = 1024$	40
USPS	9298	$16 \times 16 = 256$	10
Yale	2414	$32 \times 32 = 1024$	38
IMM	240	$480 \times 640 = 307200$	60

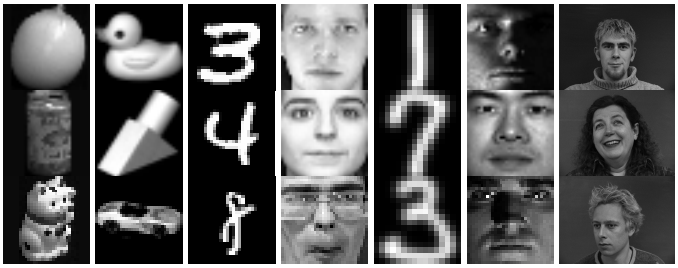


Fig. 1. Examples of the image data sets. From left to right: COIL100, COIL20, MNIST, ORL, USPS Yale, and IMM data sets.

3.2 Classification on Benchmark Data Sets

3.2.1 Data Sets

In this section, we use the following seven benchmark data sets[4] in our experiments:

- **COIL100** consists of 7200 32×32 grayscale images of 100 objects, with 72 poses for each object.
- **COIL20** consists of 1440 32×32 grayscale images of 20 objects, with 72 poses for each object.
- **MNIST** is a subset of handwritten digits from Yann LeCun's MNIST data set [20]. It consists of 4000 28×28 grayscale images of handwritten digits $0 - 9$.
- **ORL** consists of 400 32×32 grayscale face images of 40 persons. For each person, 10 images are taken at different times, varying the lighting, facial expressions and facial details.
- **USPS** consists of 9298 16×16 grayscale images of handwritten digits $0 - 9$.
- **Yale** consists of 2414 32×32 grayscale face images of 38 persons. For each person, around 60 images are taken under different illuminations.
- **IMM** data set consists of 240 640×480 grayscale face images of 40 persons. For each person, 6 images are taken with different facial expression and poses.

4. Available for download from http://www.cad.zju.edu.cn/home/dengcai/Data/data.html and https://www2.imm.dtu.dk/ aam/datasets/datasets.html

TABLE 3
Error rates (%) of the algorithms on different data sets. The best performances are bolded

	Euclidean	BoostMetric	LMNN	2DLMNN	Deep Nets	TDBML	rTDBML
COIL100	4:61 0:42	3:02 0:58	4:12 0:82	2:61 0:46	1:88 0:24	2:01 0:52	$\mathbf{1.85 \pm 0.48}$
COIL20	1:39 0:12	1:09 0:18	1:28 0:27	0:74 0:22	1:31 0:49	$\mathbf{0.15 \pm 0.12}$	0:26 0:10
MNIST	9:79 0:92	8:65 1:16	11.69 1:40	10:90 2:04	$\mathbf{4.01 \pm 0.30}$	7:29 1:12	7:55 0:96
ORL	8:94 0:59	5:99 0:78	6:03 1:25	3:73 0:61	6:59 1:24	$\mathbf{1.78 \pm 0.92}$	2:10 0:58
USPS	3:32 0:09	3:64 0:12	4:16 0:13	3:63 0:11	4:77 0:11	3:22 0:10	$\mathbf{3.13 \pm 0.12}$
Yale	31:61 2:40	15:72 3:68	12:82 3:12	9:92 4:16	$\mathbf{7.51 \pm 0.62}$	8:01 2:32	7:65 4:28
IMM	30:58 0:69	15:60 0:81	13:80 0:96	8:29 0:98	6:25 0:42	5:10 0:70	9:58 1:01

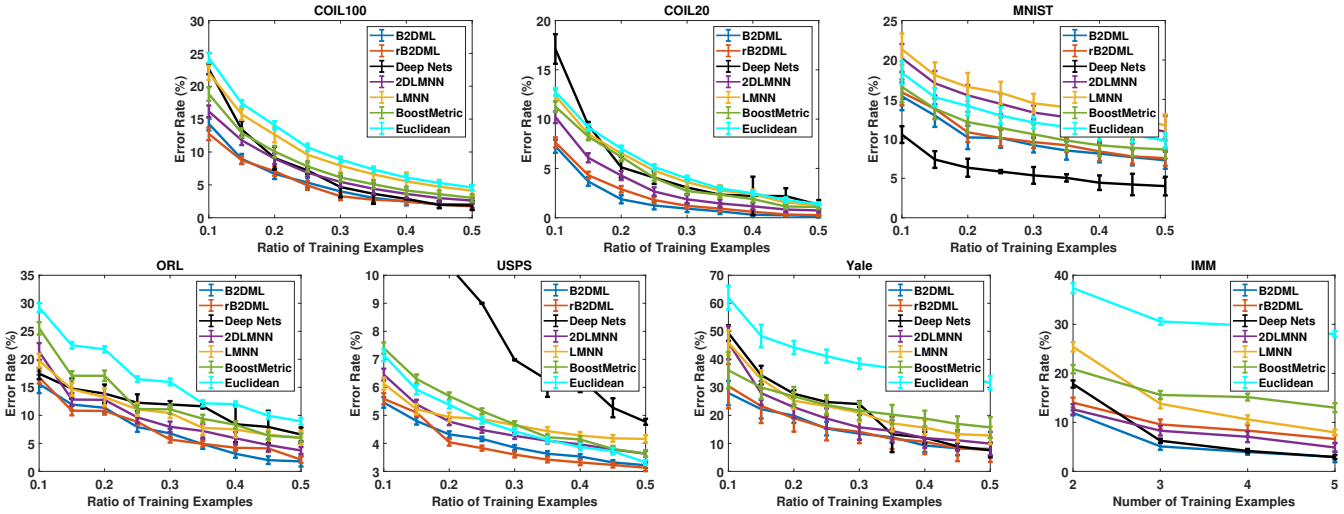


Fig. 2. Test error rates (%) of the algorithms on different data sets with different ratios of training examples.

Table 2 summarizes the statistics of the data sets. Some examples of the data sets are shown in Figure 1.

3.2.2 Performance Comparison

We evaluate TDBML and rTDBML (low-rank TDBML) against several baselines, including BoostMetric [36], large margin nearest neighbor (LMNN) [41], two dimensional large margin nearest neighbor (2DLMNN) [38], Euclidean metric, and deep nets model. For the IMM benchmark, which is a small-scale dataset with a totally of 240 images of size 640×480, we implement the AlexNet [19] as a deep model while for the rest 6 benchmarks, of whom the image size is at most 32×32, we adopted the LeNet model [20] as the deep model. The experiment details on the deep models are presented in the Supplementary materials. We first randomly generate 50/50 splits of the data between training and test set for each data set, and the average results are shown in Table 3. It can be observed that TDBML and rTDBML outperform other baselines. The overall performance of 2DLMNN is also better than that of other one-dimensional baselines, which empirically justifies our theoretical analysis of the benefits of 2DML.

To further investigate the performances of our algorithms, we also conduct experiments by varying the ratio of training examples from 10% to 50% (for IMM data set, we use varying number of training instances of each class from 2 to 5), and the results are shown in Figure 2. Still, TDBML and rTDBML can demonstrate superior performance for most scenarios, especially when only a limited number of examples are available for training. In particular, they consistently outperform BoostMetric with different ratios of training examples, especially on the COIL20, USPS, Yale, and IMM data sets. In addition, they significantly outperform 2DLMNN and LMNN on MNIST and USPS, and achieve comparable performances on the other three data sets. Compared with the deep models, TDBML and rTDBML can show better performance on the COIL20, USPS, Yale data set, and IMM data sets with a small amount of training samples (less than four). These results indicate that the TDBML and rTDBML can more effectively deal with the examples represented by matrices.

3.2.3 Running Time Comparison

In this section, we verify the efficiency of TDBML and rTDBML on a synthetic data set. We randomly generate 10 triplets of $m \times n$ matrices and vary $m = n$ from 10 to 50, and the running time of different algorithms is shown in Figure 3. It can be observed that the running time of TDBML and rTDBML (we have fixed $\Upsilon = 10$) is comparable with other baselines when the matrix size is small. As the matrix size increases, TDBML and rTDBML become become significantly faster than other algorithms, as expected. In addition, when the matrix size is small, rTDBML can be slower than TDBML as it requires extra computation cost to perform ROC to maintain a low-rank metric. As the matrix size increases, rTDBML is faster than TDBML as it can terminate the algorithm early when a new TORO matrix cannot contribute to the solution more than existing

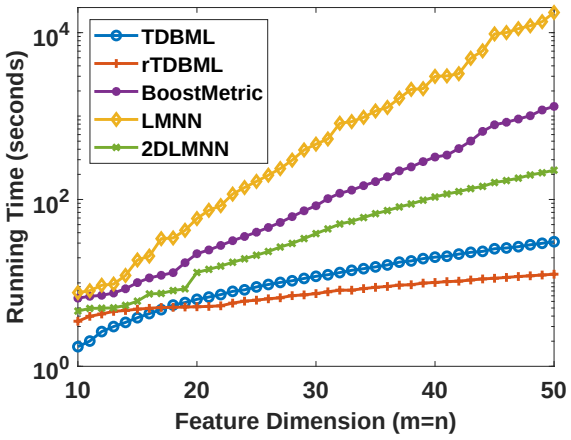


Fig. 3. Running time comparison with different algorithms.

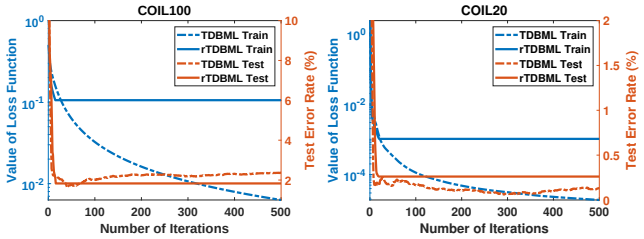


Fig. 4. Converge behavior and test error rates (%) of TDBML and rTDBML on the COIL100 and COIL20 data sets.

ones. On the other hand, although 2DLMNN also deals with matrices directly, it still has relatively high computational cost, due to its inefficient projected gradient descent.

3.2.4 Empirical Convergence Analysis

To further analyze the computational efficiency of TDBML and rTDBML, we investigate their convergence behaviors on the COIL100 and COIL20 data sets, as shown in Figure 4. It can be observed that the values of objective function of TDBML monotonically decrease with the number of iterations. In other words, as more TORO matrices are added by solving (17), we can always reduce the value of the objective function. Consequently, the algorithm only terminates when it reaches $maxIter$. On the contrary, rTDBML stops if it cannot find a new TORO matrix such that the value of the objective function decreases after ROC. As a result, it usually converges within 20 boosting iterations, and hence is much faster than TDBML. On the other hand, the test error rates of TDBML drop rapidly within in very few iterations, and become stable afterwards. In fact, increasing the number of iterations may even incur overfitting. Overall, rTDBML achieves comparable performances with TDBML with much less iterations and computation cost.

3.2.5 Parameter Sensitivity Analysis

In TDBML, we have simply kept λ_U, λ_V, $maxIter$, and K fixed, and in Section 3.2.4 have investigated the influence of number of iterations. In this section, we empirically examine the influences of other parameters.

Table 4 shows the error rates of TDBML with different values of λ_U and λ_V, where we have set $\lambda_U = \lambda_V$. It can be observed that the learning performances of TDBML are not affected by λ_U and λ_V as long as they are sufficiently small. For rTDBML, there is another tuning parameter, the rank-control parameter Υ. Figure 5 shows the test error rates of rTDBML with different values of desired rank Υ for each dimension on the COIL100 and Yale data sets. Overall, the algorithm is relatively robust to the values of Υ, though the metrics with higher rank tend to achieve better performances. The test error rates of rTDBML are low for a wide range of values of Υ as long as it is not too small.

3.3 EEG Content Decoding

The objective of this task is to decode the content of information in EEG signals, which has been actively studied in the fields such as brain computer interface (BCI) [43] and memory research [35]. In this section, we evaluate TDBML on three benchmark data sets of EEG content decoding from BCI competitions, namely, Data Set IIIa [34] from Competition III (III-IIIa), Data Set IVa [9] from Competition III (III-IVa), and Data Set IIa [26] from Competition IV (IV-IIa).

- *Data Set IIIa [34] from BCI Competition III (III-IIIa).* This data set consists of EEG signals from three subjects who performed left hand, right hand, foot, and tongue motor imagery. In this study, only the EEG signals of the left hand and right hand motor imagery are used. The signals were recorded using 60 channels, sampled at 250 Hz. The EEG signals consist of a training set and a test set, both containing 45 trials per class for subject *k3*, and 30 trials per class for subjects *k6* and *l1*.
- *Data Set IVa [9] from BCI Competition III (III-IVa).* This data set consists of the EEG signals from five subjects who performed right hand and foot motor imagery. The signals were recorded using 118 channels, sampled at 100 Hz. The signals consist of a training set and a test set, containing 280 trials in total for each subject. The numbers of training trials are 168, 224, 84, 56, 28 for subjects *aa*, *al*, *av*, *aw*, and *ay*, respectively, and the remaining trials are used as the test set.
- *Data Set IIa [26] from BCI Competition IV (IV-IIa).*This data set consists of EEG signals from nine subjects (A01 – A09) who performed left hand, right hand, foot, and tongue motor imagery. The signals were recorded using 22 channels, sampled at 250 Hz. For each subject, the EEG signals consist of a training set and a test set, both containing 72 trials per class.

Dealing with EEG signals directly by TDBML is not only computationally attractive, but also intuitively reasonable. Given a set of EEG signals $X \in \mathbb{R}^{m\times n}$, the left metric matrix $U \in \mathbb{R}^{m\times m}$ can be regarded as a spatial filter, and the right metric matrix $V \in \mathbb{R}^{n\times n}$ can be regarded as a temporal filter. In this study, we only consider rTDBML since a full-rank metric can overfit the training examples. In addition, we also augment the examples with the covariance matrix $S = XX^\top$ as it contains the information of event-related desynchronization/synchronization (ERD/ERS), which are important for classifying the cerebral activity of a specific task (e.g., imagery of hand movements).

TABLE 4
Error rates (%) of TDBML with different values of $_U$ and $_V$ ($_U$ = $_V$)

	10 ^{7}		10 ^{6}		10 ^{5}		10 ^{4}		10 ^{3}		10 ^{2}	
COIL100	2:01	0:51	2:03	0:51	2:01	0:49	2:01	0:52	2:01	0:52	2:01	0:50
COIL20	0:15	0:12	0:15	0:12	0:15	0:12	0:15	0:12	0:15	0:12	0:15	0:11
MNIST	7:29	1:12	7:29	1:11	7:29	1:12	7:29	1:12	7:29	1:12	7:31	1:24
ORL	1:78	0:92	1:78	0:92	1:78	0:92	1:78	0:92	1:78	0:92	1:78	0:98
USPS	3:22	0:10	3:22	0:10	3:22	0:10	3:22	0:10	3:22	0:10	3:24	0:11
Yale	8:01	2:32	8:01	2:32	8:01	2:32	8:01	2:32	8:01	2:32	8:04	2:01
IMM	5:05	0:52	5:05	0:52	5:10	0:70	5:10	0:70	5:10	0:70	5:14	0:76

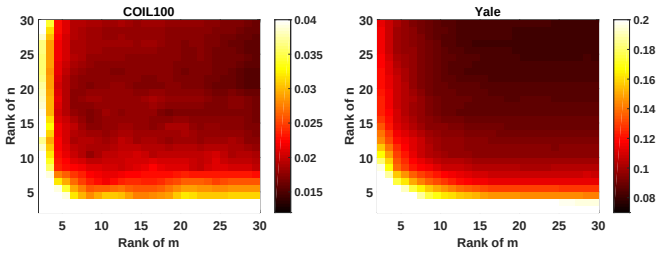


Fig. 5. Test error rates of rTDBML on COIL100 and Yale data sets with different values of .

For each data set, the EEG signals from 1 to 2 seconds after the cue instructing the subjects to perform motor imagery are used, and the data are bandpass-filtered in 8-30 Hz using 5th-order Butterworth filter, since this frequency band includes the signals involved in performing motor imagery. Consequently, the feature dimensions of the data sets are $60 \times 250 = 15000$, $118 \times 100 = 18000$ and $22 \times 250 = 5500$ respectively, which makes the problem infeasible for one-dimensional metric learning algorithms. Therefore, we consider two feature extraction methods to reduce the feature dimensions. The first one is PCA and the second one is common spatial pattern (CSP) [6], a classical feature extraction method for multichannel EEG signal analysis for BCIs. Specifically, CSP aims to find spatial filters that maximizes the projected variance ratio between the covariance matrices of two classes of brain activities. More formally, let $S_1, S_2 \in \mathbb{S}_+^C$, respectively, be the empirical covariance matrices of class 1 and class 2, the objective of CSP is to solve the following simultaneous diagonalization problem:

$$\max_{W \in \mathbb{R}^m} \ _s \text{trace}\,(\Lambda)$$
$$\text{s.t. } W^\top X_1^\top X_1 W = \Lambda,\ W^\top X_2^\top X_2 W = I,$$

where s is the number of spatial filters, $\Lambda \in \mathbb{R}^{s \times s}$ is a diagonal matrix, and I is the identity matrix. In the experiments, we set $s = 3$, as suggested in [23]. After dimensionality reduction using PCA or CSP, the features are classified using k-NN performed with one of the following metrics: Euclidean, BoostMetric, and LMNN. For comparison purpose, only the EEG signals from the left hand and right hand motor imagery are used, since CSP is only designed to extract the spatial features for binary classification problems.

Table 5 reports the average classification accuracies of different algorithms. Overall, PCA cannot extract useful features for EEG classification, and therefore the algorithms based on PCA have much lower accuracies than other algorithms. On the other hand, as a supervised feature extraction method, CSP learns spatial filters that maximize the projected variance ratio between the covariance matrices of two classes of brain activities. As a result, it significantly improves the learning performances. rTDBML is also a supervised learning algorithm. More important, it learns not only spatial filters, but also temporal filters. As a result, it may extract more informative features for EEG classification. We observe that rTDBML outperforms other baselines on 11 out of 17 subjects, and achieve the highest average classification accuracy, yielding an average improvement of 14.55% over PCA+k-NN, and of 1.86% over the second best algorithm (i.e., CSP+BoostMetric).

We further investigate the effectiveness of rTDBML by varying the ratio of training examples. For each competition data set, we first put the training and test trials together, and then randomly split data into training and test set with different ratios. For each ratio, we repeat the process 10 times and the average test error rates with standard deviations are reported in Figure 6. It can be observed that rTDBML outperforms other baselines, especially on the data sets III-IIIa and III-IVa. Compared with results on the image data sets, however, the improvements are not significant. We speculate that the reason is two-fold: **1.** The baseline algorithms compared with in Figure 6 are based on CSP, which has been proven a powerful approach to extracting spatial features and reducing dimensionality of EEG signals [23]; **2.** In BCI, spatial information is more important than temporal information for EEG content decoding [6], [31]. Nevertheless, Figure 6 still indicates that the overall content decoding performances still benefit from both spatial and temporal information extracted by rTDBML.

4 Conclusion

In this paper, we theoretically justify the benefits of 2DML by showing that its Rademacher complexity has a faster convergence rate than 1DML. To the best of our knowledge, this is the first time in the literature that the benefits of a two dimensional learning algorithm have been systematically studied. In addition, we also present TDBML, an efficient algorithm for two dimensional metric learning of matrix data. Instead of vectorizing the matrix data as in classical metric learning algorithms, TDBML works directly on the data in matrix representation, and therefore scales well the feature dimensions. In addition, we also propose a method to explicitly control the rank of the metric matrices by rank-one correction. Empirical evaluation on both image

TABLE 5
Classification accuracy (%) of different algorithms on EEG content decoding data sets. The best performances are bolded.

Data Set	Subject	PCA +Eculidean	PCA +BoostMetric	PCA +LMNN	CSP +Eculidean	CSP +BoostMetric	CSP +LMNN	rTDBML
III-IIIa	*k3*	77:78	81:11	84:44	94:44	96:67	**98.89**	**98.89**
	k6	51:11	61:67	61:11	63:33	62:22	67:78	**71.67**
	l1	81:11	88:89	84:44	93:33	96:67	93:33	**98.33**
	Average	70:00	77:22	76:66	83:70	85:19	86:67	**89.63**
III-IVa	*aa*	57:14	62:50	60:71	66:07	71:43	66:96	**76.79**
	al	83:93	87:50	89:29	94:90	94:64	94:90	**96.43**
	av	47:45	45:41	47:45	53:57	**54.08**	52:04	52:04
	aw	66:96	68:75	63:84	71:43	**80.35**	77:68	79:02
	ay	56:75	49:60	57:54	51:59	63:10	67:46	**70.63**
	Average	62:45	62:75	63:76	67:51	72:72	71:81	**74.98**
IV-IIa	*A01*	76:39	77:78	79:86	86:11	90:28	**91.67**	**91.67**
	A02	51:39	52:08	52:08	53:47	**56.25**	54:86	55:56
	A03	81:25	88:89	91:67	**96.53**	95:84	**96.53**	95:84
	A04	60:42	65:28	67:37	72:92	70:84	71:53	**73.61**
	A05	54:86	52:08	49:60	54:17	60:42	61:81	**63.89**
	A06	60:42	68:06	65:28	**72.92**	71:53	70:14	70:84
	A07	63:89	68:06	69:45	79:17	81:25	79:86	**82.64**
	A08	70:14	74:31	72:92	**93.75**	92:37	92:37	91:67
	A09	75:00	81:25	81:25	90:28	**93.75**	92:37	**93.75**
	Average	65:97	69:75	69:94	77:70	79:17	79:02	**79.94**
Overall average		65:65	69:01	69:31	75:76	78:33	78:25	**80.19**

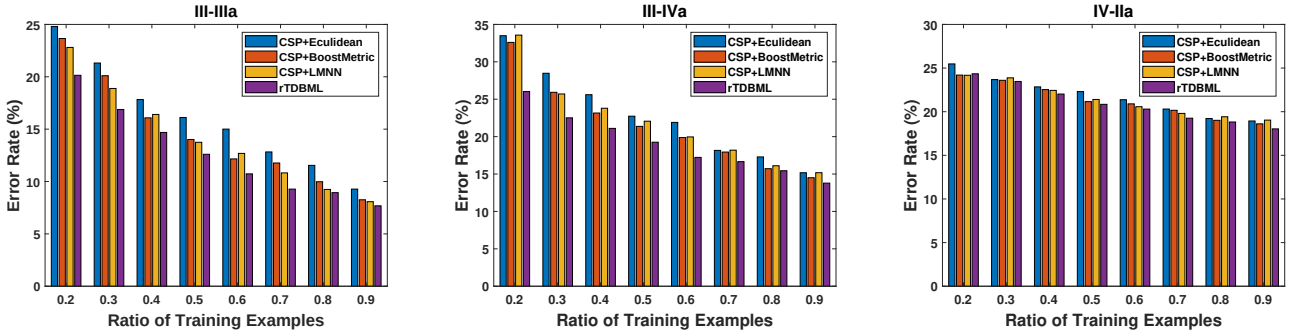


Fig. 6. Test error rates (%) of the algorithms on different data sets with different ratios of training examples.

classification and EEG content decoding shows that TDBML algorithms outperform the baselines in terms of both accuracy and scalability.

In the future, we are particularly interested in the further study of the application of TDBML on EEG signal analysis. First, it would be interesting to further investigate the spatial and temporal filters learned by TDBML, as they could possibly give some neurophysiologically meaningful explanation of the metric matrices. In addition, as we learn a low rank matrices U and V, the EEG signals are actually projected into lower dimensional space by a bilinear transformation. The compressed EEG segment could be regarded as a *signature* that captures the essence of the signal. A further look at these compressed signals may reveal some insights into the brain activities. In addtion, besides the explicit rank control, we are also interested in other regularization approaches to control the model complexity and incorporate prior knowledge. Finally, we are also interested in extending our theoretical analysis to other two-dimensional or bilinear algorithms, such as PCA [45], support vector machines [32], logistic regression [39], and deep metric learning [18].

ACKNOWLEDGEMENT

The authors gratefully acknowledge financial support from the Science and the Natural Sciences and Engineering Research Council of Canada (NSERC), Discovery Grants Program, the Faculty of Science at the University of Western Ontario, the China Scholarship Council, and the Science and Technology Development Fund, Macau SAR (File no. 0045/2019/AFJ and 0018/2019/AKP).

REFERENCES

[1] M. Barnathan, V. Megalooikonomou, C. Faloutsos, S. Faro, and F. B. Mohamed, "TWave: High-order analysis of functional MRI," *Neuroimage*, vol. 58, no. 2, pp. 537–548, 2011.

[2] P. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2002.

[3] A. Bellet and A. Habrard, "Robustness and generalization for metric learning," *Neurocomputing*, vol. 151, pp. 259–267, 2015.

[4] A. Bellet, A. Habrard, and M. Sebban, "Similarity learning for provably accurate sparse linear classification," in *Proceedings of the International Conference on Machine Learning*, 2012, pp. 1491–1498.

[5] J. C. Bezdek and R. J. Hathaway, "Convergence of alternating optimization," *Neural, Parallel and Scientific Computations*, vol. 11, pp. 351–368, December 2003.

[6] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Müller, "Optimizing spatial filters for robust eeg single-trial analysis," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 41–56, 2008.

[7] Q. Cao, Z.-C. Guo, and Y. Ying, "Generalization bounds for metric and similarity learning," *Machine Learning*, vol. 102, no. 1, pp. 115–132, 2016.

[8] C. Christoforou, R. Haralick, P. Sajda, and L. C. Parra, "Second-order bilinear discriminant analysis," *Journal of Machine Learning Research*, vol. 11, pp. 665–685, 2010.

[9] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Muller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 993–1002, 2004.

[10] M. Dyrholm, C. Christoforou, and L. C. Parra, "Bilinear discriminant component analysis," *Journal of Machine Learning Research*, vol. 8, pp. 1097–1111, 2007.

[11] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. R. Salakhutdinov, "Neighbourhood components analysis," in *Advances in Neural Information Processing Systems*, 2005, pp. 513–520.

[12] N. Golowich, A. Rakhlin, and O. Shamir, "Size-independent sample complexity of neural networks," in *Conference On Learning Theory*, 2018, pp. 297–299.

[13] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*. MIT press Cambridge, 2016.

[14] Z.-C. Guo and Y. Ying, "Guaranteed classification via regularized similarity learning," *Neural Computation*, vol. 26, no. 3, pp. 497–522, 2014.

[15] C. Hou, F. Nie, D. Yi, and Y. Wu, "Efficient image classification via multiple rank regression," *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 340–352, 2012.

[16] C. Hou, F. Nie, C. Zhang, D. Yi, and Y. Wu, "Multiple rank multilinear SVM for matrix data classification," *Pattern Recognition*, vol. 47, no. 1, pp. 454–469, 2014.

[17] 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.

[18] M. Kaya and H. Ş. Bilge, "Deep metric learning: A survey," *Symmetry*, vol. 11, no. 9, p. 1066, 2019.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] S. H. Lee and S. Choi, "Two-dimensional canonical correlation analysis," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 735–738, 2007.

[22] Y. Lei, S.-B. Lin, and K. Tang, "Generalization bounds for regularized pairwise learning." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018, pp. 2376–2382.

[23] F. Lotte and C. Guan, "Regularizing common spatial patterns to improve bci designs: unified theory and new algorithms," *IEEE Transactions on biomedical Engineering*, vol. 58, no. 2, pp. 355–362, 2011.

[24] M. E. Lübbecke and J. Desrosiers, "Selected topics in column generation," *Operations Research*, vol. 53, no. 6, pp. 1007–1023, 2005.

[25] J. Ludeña-Choez, R. Quispe-Soncco, and A. Gallardo-Antolín, "Bird sound spectrogram decomposition through non-negative matrix factorization for the acoustic classification of bird species," *PloS One*, vol. 12, no. 6, p. e0179403, 2017.

[26] M. Naeem, C. Brunner, R. Leeb, B. Graimann, and G. Pfurtscheller, "Seperability of four-class motor imagery data using independent components analysis," *Journal of Neural Engineering*, vol. 3, no. 3, pp. 208–216, 2006.

[27] F. Nie, S. Xiang, Y. Song, and C. Zhang, "Extracting the optimal dimensionality for local tensor discriminant analysis," *Pattern Recognition*, vol. 42, no. 1, pp. 105–114, 2009.

[28] F. Nie, H. Zhang, R. Zhang, and X. Li, "Robust multiple rank-*k* bilinear projections for unsupervised learning," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2574–2583, 2018.

[29] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Non-negative multilinear principal component analysis of auditory temporal modulations for music genre classification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 576–588, 2010.

[30] M. Perrot and A. Habrard, "A theoretical analysis of metric hypothesis transfer learning," in *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1708–1717.

[31] G. Pfurtscheller and C. Neuper, "Motor imagery and direct brain-computer communication," *Proceedings of the IEEE*, vol. 89, no. 7, pp. 1123–1134, 2001.

[32] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Bilinear classifiers for visual recognition," in *NIPS*, 2009, pp. 1482–1490.

[33] S. Rosset, J. Zhu, and T. Hastie, "Boosting as a regularized path to a maximum margin classifier," *Journal of Machine Learning Research*, vol. 5, pp. 941–973, 2004.

[34] A. Schlögl, F. Lee, H. Bischof, and G. Pfurtscheller, "Characterization of four-class motor imagery EEG data for the BCI-competition 2005," *Journal of Neural Engineering*, vol. 2, no. 4, pp. L14–L22, 2005.

[35] M. Schönauer, S. Alizadeh, H. Jamalabadi, A. Abraham, A. Pawlizki, and S. Gais, "Decoding material-specific memory reprocessing during sleep in humans," *Nature Communications*, vol. 8, 2017.

[36] C. Shen, J. Kim, L. Wang, and A. Hengel, "Positive semidefinite metric learning with boosting," in *Advances in Neural Information Processing Systems*, 2009, pp. 1651–1659.

[37] C. Shen, J. Kim, L. Wang, and A. v. d. Hengel, "Positive semidefinite metric learning using boosting-like algorithms," *Journal of Machine Learning Research*, vol. 13, pp. 1007–1036, 2012.

[38] K. Song, F. Nie, and J. Han, "Two dimensional large margin nearest neighbor for matrix classification." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2017, pp. 2751–2757.

[39] K. Song, F. Nie, J. Han, and X. Li, "Rank-*k* 2-D Multinomial Logistic Regression for Matrix Data Classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3524–3537, 2018.

[40] R. Tomioka and K.-R. Müller, "A regularized discriminative framework for EEG analysis with application to brain–computer interface," *NeuroImage*, vol. 49, no. 1, pp. 415–432, 2010.

[41] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *NIPS*, 2005, pp. 1473–1480.

[42] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[43] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

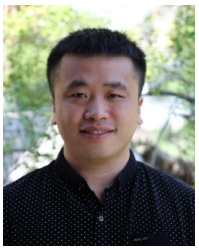
[44] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in *NIPS*, 2003, pp. 521–528.

[45] J. Ye, "Generalized low rank approximations of matrices," *Machine Learning*, vol. 61, no. 1-3, pp. 167–191, 2005.

[46] J. Ye, R. Janardan, and Q. Li, "Two-dimensional linear discriminant analysis," in *NIPS*, 2005, pp. 1569–1576.

[47] L. Yu, Y. Gao, J. Zhou, and J. Zhang, "Parameter-efficient deep neural networks with bilinear projections," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[48] S.-H. Zhong, Y. Liu, and Y. Liu, "Bilinear deep learning for image classification," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 343–352.



Di Wu is currently an Adjunct Professor at McGill University. Before joining Samsung, he did postdoctoral research at Montreal MILA and Stanford University. He received the Ph.D. degree from McGill University, Montreal, Canada, in 2018 and the MSc degree from Peking University, Beijing, China, in 2013. Di's research interests mainly lie in designing algorithms (e.g., Reinforcement learning, operation research) for sequential decision-making problems and data-efficient machine learning algorithms (e.g., Transfer Learning, Meta-Learning, and Multitask Learning). He is also interested in leveraging such algorithms for applications in real-world systems (e.g., Communication Systems, Smart Grid, and Intelligent Transportation Systems).