

Research Article

A Fast Logdet Divergence Based Metric Learning Algorithm for Large Data Sets Classification

Jiangyuan Mei,¹ Jian Hou,² Jicheng Chen,¹ and Hamid Reza Karimi³

¹ Research Institute of Intelligent Control and Systems, Harbin Institute of Technology, Harbin 150080, China

² School of Information Science and Technology, Bohai University, No. 19, Keji Road, Jinzhou 121013, China

³ Department of Engineering, Faculty of Engineering and Science, University of Agder, 4898 Grimstad, Norway

Correspondence should be addressed to Jiangyuan Mei; meijiangyuan@gmail.com

Received 17 March 2014; Revised 2 April 2014; Accepted 4 May 2014; Published 18 May 2014

Academic Editor: Shen Yin

Copyright © 2014 Jiangyuan Mei et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Large data sets classification is widely used in many industrial applications. It is a challenging task to classify large data sets efficiently, accurately, and robustly, as large data sets always contain numerous instances with high dimensional feature space. In order to deal with this problem, in this paper we present an online Logdet divergence based metric learning (LDML) model by making use of the powerfulness of metric learning. We firstly generate a Mahalanobis matrix via learning the training data with LDML model. Meanwhile, we propose a compressed representation for high dimensional Mahalanobis matrix to reduce the computation complexity in each iteration. The final Mahalanobis matrix obtained this way measures the distances between instances accurately and serves as the basis of classifiers, for example, the k -nearest neighbors classifier. Experiments on benchmark data sets demonstrate that the proposed algorithm compares favorably with the state-of-the-art methods.

1. Introduction

Recently, large data sets classification has become one of the hottest research topics since it is the building block in many industrial and computer vision applications, such as fault diagnosis in complicated systems [1, 2], automated optical inspection for complex workpieces [3], and face recognition in large-capacity databases [4]. In these large data sets, there are usually numerous instances represented in high dimensional feature spaces, which makes the problem of large data sets classification very difficult.

There are various classification algorithms which have been intensively explored, including Fisher's linear discriminant, support vector machines, and k -nearest neighbor. However, these methods all rely on measuring the distance over the multidimensional feature space of instances accurately and robustly. Traditional distance metrics, including Euclidean and $L1$ distance, usually assign equal weights to all features and ignore the difference among these features, which is not practical in the real applications. In fact, these features may have different relevance to the category of

instances. Some of them have strong correlation with the label of instances while others have weak or no correlation. Therefore, an appropriate distance or similarity metric which can build the relationship between feature space and category of instances should be learned to measure the divergence among instances. Metric learning is a popular approach to accomplish such a learning process. In this paper we select Mahalanobis distance as the distance metric between instances.

The Mahalanobis distance is a standard distance metric parameterized by a positive semidefinite (PSD) matrix M . Given a data set $\{x_i\}$, with $x_i \in \mathfrak{R}^d$, $i = 1, 2, \dots, n$, the square Mahalanobis distance between instances x_i and x_j is defined as

$$d_M(x_i, x_j) = (x_i - x_j)^T M (x_i - x_j). \quad (1)$$

The Mahalanobis distance satisfies all the conditions of metric definitions, including (1) nonnegativity, $d_M(x_i, x_j) \geq 0$; (2) symmetry, $d_M(x_i, x_j) = d_M(x_j, x_i)$; (3) triangle inequality, $d_M(x_i, x_j) + d_M(x_j, x_k) > d_M(x_i, x_k)$; and (4) identity

of indiscernibles, $d_M(x_i, x_j) = 0$ if only $x_i = x_j$. In the case that $M = I$, where I is an identity matrix, the Mahalanobis distance degenerates to the Euclidean distance. Mahalanobis distance has some considerable advantages over other metrics. Firstly, the Mahalanobis distance is scale invariant, which means that the scale of the Mahalanobis distance has no influence on the performance of classification or clustering. Secondly, this metric takes into account the correlations of different features. In general, the element of the off-diagonal element in Mahalanobis matrix is not zero, which helps build a more accurate relationship among instances. When we apply singular value decomposition to the Mahalanobis matrix, it can be decomposed as $M = H\Sigma H^T$. Here, H is a unitary matrix which satisfies $HH^T = I$, where left unitary matrix is the transpose of right unitary matrix due to the symmetry of Mahalanobis matrix M . And Σ is a diagonal matrix which contains all the singular values. Thus, the square Mahalanobis distance can be rewritten as

$$\begin{aligned} d_M(x_i, x_j) &= (x_i - x_j)^T H\Sigma H^T (x_i - x_j) \\ &= (H^T x_i - H^T x_j)^T \Sigma (H^T x_i - H^T x_j). \end{aligned} \quad (2)$$

From (2) we can see that the Mahalanobis distance has two main functions. The first one is to find the best orthogonal matrix H to remove the couplings among features and build new features. The second one is to assign weights Σ to the new feature. These two functions enable Mahalanobis distance to measure the distance between instances effectively.

Learning such a Mahalanobis distance is a complex procedure. Several classical metric learning algorithms such as probabilistic global distance metric learning (PGDM) [5], large margin nearest neighbor (LMNN) [6], and information-theoretic metric learning (ITML) [7] have been proposed to learn the Mahalanobis distance. However, these algorithms seem computationally inefficient for large data sets, and it is a hard nut to accelerate the learning process with large data sets. In practice, there are two main challenges in scalability for large data sets. The first one is that the data sets may contain thousands of instances. To avoid local minimum in the metric learning process, as many as possible useful instances should be used in training. This leads to low computation efficiency in metric learning. The second one is that the dimensionality of the data may be very large. The number of parameters involved in the metric learning problem is $O(\min(n^2, d^2))$, where n is the number of training instances and d is the dimensionality of the data. Thus, the running time for training Mahalanobis distance would be quadratic dependent on the number of dimensions. At the same time, estimating a quadratic number of parameters would also pose a new challenge [8].

In dealing with the challenge from numerous instances, we find online metric learning as a good solution. Online metric learning methods have two major advantages over traditional offline methods. First, in many practical applications, the system can only receive several instances or constraints at a time, and the desired Mahalanobis distance should be updated gradually over time. For example, in a process control system [9–11], various sensors are utilized

to collect a group of feature data at one time, which may influence the Mahalanobis distance used in detecting fault. In this situation, online metric learning can be used to address the need of Mahalanobis distance updating. Second, some offline applications with numerous instances can be converted to online metric learning problems. Compared with offline learning, online metric learning reduces the running time dramatically, as the Mahalanobis distance is optimized step by step rather than calculated at a time. There are some online metric learning methods in the literatures, including pseudometric online learning algorithm (POLA) [12], online ITML algorithm [7], and Logdet exact gradient online metric learning algorithm (LEGO) [13]. However, existing methods usually suffer from a number of drawbacks. The POLA involves a eigenvector extraction process in each step, which means a large computation load, especially with a high dimensional feature space. Although online ITML is faster than POLA, its improvement in computation efficiency is accompanied by loss in performance as the loss bounds of online ITML are dependent on the training data. The LEGO improves on the online ITML and achieves both high precision and fast speed at the same time. However, in the case of high dimensional feature space, LEGO fails to reduce the computational complexity at each step effectively. Thus, LEGO algorithm cannot well solve the problem illustrated in the second challenge.

To address the challenges and opportunities raised by larger data sets, this paper proposes a new metric learning strategy. First of all, we describe a novel online Logdet divergence based metric learning model which uses triplets as the training constraints. This model is shown to perform better than traditional metric learning algorithms in both precision and robustness. Then, to reduce the computational complexity in each iteration, a compressed representation for high dimensional Mahalanobis matrix is proposed. A low-rank Mahalanobis matrix is utilized to represent the original high dimensional Mahalanobis matrix in the metric learning process. As a result, the proposed algorithm solves the problems raised by numerous instances as well as high dimensional feature space.

The remainder of this paper is organized as follows. In Section 2, the proposed online Logdet divergence based metric learning model is presented. Then, the method of compressed representation for high dimensional Mahalanobis matrix is described in Section 3. Section 4 reports the experimental results on UCI machine learning repository to demonstrate the effectiveness of the proposed algorithm. Finally, we draw conclusions and point out future directions in Section 5.

2. Online Logdet Divergence Based Metric Learning Model

In the metric learning process, most successful results rely on having access to all useful instances or constraints in the whole data set. However, in some real applications, we cannot obtain all the instances at one time because of some reasons. For example, if there are too many instances in the

training sets, reading in all the data may be out of memory of computer. Another example is that some online applications only provide several instances or constraints at one time. Therefore, we should desire a metric learning model which can update the Mahalanobis distance gradually as instances or constraints are received. Thus, our metric learning framework is to solve the following iterative minimization problem:

$$M_{t+1} = \arg \min_{M \geq 0} D_{\text{ld}}(M, M_t) + \eta_t \ell(M), \quad (3)$$

where $\eta_t > 0$ is a regularization parameter which balances the regularization function $D(M, M_t)$ and loss function $\ell(M)$.

In this framework, the first item $D_{\text{ld}}(M, M_t)$ is a regularization function which is used to guarantee the stability of metric learning process. The function $D_{\text{ld}}(\cdot)$ represents Logdet divergence [14]:

$$D_{\text{ld}}(M, M_t) = \text{tr}(MM_t^{-1}) - \log(\det(MM_t^{-1})) - d, \quad (4)$$

where d is the dimension of M . There are several advantages when using Logdet divergence to regularize the metric learning process. First, the Logdet divergence between the covariance matrices is equivalent to the Kullback-Leibler divergence between corresponding multivariate Gaussian distributions [15]. Second, the Logdet divergence is general linear group transformation invariant, that is, $D_{\text{ld}}(M, M_t) = D_{\text{ld}}(S^TMS, S^TM_tS)$, where S is an invertible matrix [7]. These desirable properties make Logdet divergence very useful in metric learning and the proposed algorithm in this paper is called Logdet divergence based metric learning (LDML).

The second item in the framework $\ell(M)$ is the loss function measuring the loss between prediction distance \hat{y}_t and target distance y_t at time step t . Obviously, when the total loss function $L(M) = \sum_t \ell(\hat{y}_t, y_t)$ reaches its minimal, the obtained M is the most close to the desired distance function. There are several methods to choose the prediction distance \hat{y}_t and target distance y_t . In the proposed framework, we select triplet $\{x_i, x_j, x_k\}$, which represents the instance x_i is more similar to the instance x_j than instance x_k , as the labels of the training samples. The prediction distance is $\hat{y}_t = d_{M_t}(x_i^t, x_k^t) - d_{M_t}(x_i^t, x_j^t)$ and the target distance is chosen as $y_t = \rho$. Thus the corresponding loss function is expressed as

$$\ell(M_t) = \max(0, \rho + d_{M_t}(x_i^t, x_j^t) - d_{M_t}(x_i^t, x_k^t)). \quad (5)$$

When receiving a new triplet $\{x_i^t, x_j^t, x_k^t\}$ at time step t , if $d_{M_t}(x_i^t, x_k^t) - d_{M_t}(x_i^t, x_j^t) \geq \rho$, there is no loss when using the current M_t to represent the relationship among these three instances; if $d_{M_t}(x_i^t, x_k^t) - d_{M_t}(x_i^t, x_j^t) < \rho$, the current M_t should be updated to a better Mahalanobis distance to reduce the loss.

In this formulation, the triplet constraints $\{x_i, x_j, x_k\}$ which represent proximity relationships are used as constraints. In online ITML and LEGO algorithms, they all use pairwise constraints as training samples. If (x_i, x_j) belongs to the same category, the obtained Mahalanobis distance should satisfy $d_{M_t}(x_i, x_j) < u$, where u is a desired superior limit of distance among instances in the same category; if (x_i, x_k)

are dissimilar, the constraints for Mahalanobis distance M are $d_{M_t}(x_i, x_k) > v$, where v is a desired lower limit of distance among instances in the different categories. Although the pairwise constraints are weaker than the class labels [16], they are still stronger than triplet constraints. The reason is obvious, the distributions and instances quantities are different in different categories, but the desired superior limit u and lower limit v are the same for every category. Thus the Mahalanobis distance learned using online ITML and LEGO algorithms would get conservative results in this situation. The work [17] has pointed out that triplet constraints can be derived from pairwise constraints, but not vice versa. Therefore, the triplet constraint is weaker as well as more natural than pairwise constraints. And the corresponding online LDML algorithm can achieve more accurate results than online ITML and LEGO algorithms.

3. Compressed Representation for High Dimensional Mahalanobis Matrix

Although the online metric learning model can avoid semidefinite programming and reduce the amount of computations sharply, the computation complexity is restricted by the dimensionality of the feature space. As mentioned above, the number of parameters in Mahalanobis matrix is quadratic to the dimensionality d . A Mahalanobis matrix with large number of parameters will lead to an inefficient computation in the metric learning process. To address this problem, we use compressed representations [8] method to learn, store, and evaluate the Mahalanobis matrix efficiently. The Mahalanobis distance function M with a full $d \times d$ matrix is constrained as the sum of a high dimensional identity I^d plus a low-rank symmetric matrix M_L , expressed as

$$M = I^d + M_L = I^d + ULU^T, \quad (6)$$

where $U \in R^{d \times k}$ is orthogonal basis and $U \in R^{k \times k}$ is a symmetric matrix with $k \ll \min(n, d)$. Correspondingly, the Mahalanobis distance function at time step t can be decomposed as $M_t = I^d + UL_tU^T$.

Theorem 1. $D_{\text{ld}}(M, M_t) = D_{\text{ld}}(F, F_t)$, where $F = I^k + L$ and $F_t = I^k + L_t$.

Proof. First of all, we consider the first item in (4):

$$\begin{aligned} & \text{tr}(MM_t^{-1}) \\ &= \text{tr}\left(\left(I^d + ULU^T\right)\left(I^d + UL_tU^T\right)^{-1}\right) \\ &= \text{tr}\left(\left(I^d + ULU^T\right)\left(I^d - U\left(I^k - (L_t + I^k)^{-1}\right)U^T\right)\right) \\ &= \text{tr}\left(I^d + ULU^T\right) \\ &\quad - \text{tr}\left(\left(I^d + ULU^T\right)U\left(I^k - F_t^{-1}\right)U^T\right) \\ &= \text{tr}\left(I^k\right) + d - k + \text{tr}\left(LU^TU\right) \\ &\quad - \text{tr}\left(U^T\left(I^d + ULU^T\right)U\left(I^k - F_t^{-1}\right)\right) \end{aligned}$$

$$\begin{aligned}
&= \text{tr}(F) + d - k - \text{tr}(I^k - F_t^{-1}) \\
&\quad - \text{tr}(L(I^k - F_t^{-1})) \\
&= \text{tr}(F) - \text{tr}(F(I^k - F_t^{-1})) + d - k \\
&= \text{tr}(FF_t^{-1}) + d - k,
\end{aligned} \tag{7}$$

where the second equality follows from the fact of Woodbury matrix identity

$$(A + ECF)^{-1} = A^{-1} - A^{-1}E(C^{-1} + FA^{-1}E)FA^{-1}, \tag{8}$$

and the third equality follows from the fact that $\text{tr}(AB) = \text{tr}(BA)$.

Then, the second item in (4) can be converted as

$$\begin{aligned}
&\log(\det(MM_t^{-1})) \\
&= \log(\det((I^d + ULU^T)(I^d + UL_tU^T)^{-1})) \\
&= \log\left(\frac{\det(I^d + ULU^T)}{\det(I^d + UL_tU^T)}\right) \\
&= \log\left(\frac{\det(I^k + L)}{\det(I^k + L_t)}\right) \\
&= \log(\det(FF_t^{-1})),
\end{aligned} \tag{9}$$

where the second equality follows from the fact that $\det(AB^{-1}) = \det(A)/\det(B)$, and the third equality follows from the fact that $\det(I^m + AB) = \det(I^m + BA)$ for all $A \in R^{m \times n}$ and $B \in R^{n \times m}$.

Thus, we can get the following equation:

$$\begin{aligned}
&D_{\text{ld}}(M, M_t) \\
&= \text{tr}(MM_t^{-1}) - \log(\det(MM_t^{-1})) - d \\
&= \text{tr}(FF_t^{-1}) + d - k + \log(\det(FF_t^{-1})) - d \\
&= D_{\text{ld}}(F, F_t),
\end{aligned} \tag{10}$$

hence proved. \square

From Theorem 1 we can see, if we build a relationship between M and F using the orthogonal basis U , learning a low dimensional symmetric matrix F is equal to learning the original Mahalanobis distance function M . The advantage of this method is that the computational complexity of each iteration will decrease significantly. Thus, it deserves obtaining the updating formulation of F_t to evaluate the true Mahalanobis distance function M_t .

Assume that $\bar{X} = U^T X$ represents the reduced-dimensional data under the orthogonal basis U ; then we can get the corresponding variables $\bar{p}_t = U^T(x_i^t - x_j^t) = U^T p_t$

and $\bar{q}_t = U^T(x_i^t - x_k^t) = U^T q_t$. Thus, the loss function can be rewritten as

$$\ell(F) = \bar{p}_t F \bar{p}_t^T - \bar{q}_t F \bar{q}_t^T + \bar{\rho}, \tag{11}$$

where $\bar{\rho} = p_t p_t^T - \bar{p}_t \bar{p}_t^T - q_t q_t^T + \bar{q}_t \bar{q}_t^T + \rho$. The function $D(F, F_t) + \eta_t \ell(F)$ reaches its minimum when its gradient is zero. Thus, we get the following equation by setting gradient of (3) to be zero with respect to F :

$$F_{t+1} = (F_t^{-1} + \eta_t (\bar{p}_t \bar{p}_t^T - \bar{q}_t \bar{q}_t^T))^{-1}. \tag{12}$$

Since matrix inverse is computationally very expensive, in order to avoid inverse, we apply the Sherman-Morrison inverse formula to solve (12). The standard Sherman-Morrison formula is

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1}u}. \tag{13}$$

However, in our updating equation, there are two items which are the outer product of vectors. To solve this problem, we assume that $\Gamma_t = (F_t^{-1} + \eta_t p_t p_t^T)^{-1}$, and (12) is split into two standard Sherman-Morrison inverse questions:

$$\begin{aligned}
\Gamma_t &= (F_t^{-1} + \eta_t \bar{p}_t \bar{p}_t^T)^{-1}, \\
F_{t+1} &= (\Gamma_t^{-1} - \eta_t \bar{q}_t \bar{q}_t^T)^{-1}.
\end{aligned} \tag{14}$$

Applying the Sherman-Morrison formula, we arrive at an analytical expression for F_{t+1}

$$\begin{aligned}
\Gamma_t &= F_t - \frac{\eta_t F_t \bar{p}_t \bar{p}_t^T F_t}{1 + \eta_t \bar{p}_t^T F_t \bar{p}_t}, \\
F_{t+1} &= \Gamma_t + \frac{\eta_t \Gamma_t \bar{q}_t \bar{q}_t^T \Gamma_t}{1 - \eta_t \bar{q}_t^T \Gamma_t \bar{q}_t}.
\end{aligned} \tag{15}$$

The corresponding Mahalanobis distance function is evaluated as $M_t = I^d + U(F_t - I^k)U^T$. Using the compressed representations technique, the computational complexity reduces from $O(\min(n^2, d^2))$ to $O(k^2)$ per iteration. At the same time, the storage space in the metric learning process also reduces sharply.

There are several practical methods to build the basis [18]; one of the most efficient methods is to choose the k first left singular vector after applying the singular value decomposition (SVD) to the original training data X :

$$X = R \Sigma S^T, \tag{16}$$

where R and S are left and right unitary matrix. And the orthogonal basis U is selected as

$$U = [r_1 \ r_2 \ \cdots \ r_k]. \tag{17}$$

This method is simple but time consuming. The computational complexity of singular value decomposition is $O(n^2 d + n d^2 + d^3)$ when $d \ll n$. In large data set, the objects

TABLE 1: Data sets used in the experiments.

Name	Number of attributes	Number of classes	Number of instances
Iris	4	3	150
Seeds	7	3	210
Wine	13	3	178
WDBC	30	3	569
Sonar	60	2	208
Ionosphere	3	2	351
Letter-Recognition	16	26	20000
Spambase	57	2	4601
Isolet	617	26	7797
Semeion	256	10	1593
Mfeat	649	10	2600

are always with high d and n . Thus, using the traditional SVD method will lead to enormous computation. Moreover, in this online metric learning model, the instances cannot be obtained at the orthogonal basis building step, which is regarded as a preprocess of the proposed method. Therefore, an online SVD algorithm should be introduced in our framework. We use a truncated incremental SVD [19, 20] to obtain the basis in our algorithm, which can decline the computational complexity to $O(nk^3)$; when $k \ll d$, the truncated incremental SVD algorithm will sharply reduce the computation time compared with traditional SVD algorithm.

There are two main constraints for the regularization parameter η_t . First, it is used to make sure that F_{t+1} is a PSD matrix in each iteration. When $0 < \eta_t < 1/\text{tr}((I - M_t)^{-1}M_t\bar{q}_t\bar{q}_t^T)$, the F_{t+1} will be a PSD matrix if F_t is a PSD matrix. This satisfies the first constraint. Second, it also controls the balance of the regularization function and the loss function. In this paper, we select $\eta_t = \alpha/\text{tr}((I - M_t)^{-1}M_t\bar{q}_t\bar{q}_t^T)$, where α is the learning rate parameter which is chosen between 0 and 1. On one hand, if α is too large, the F_{t+1} will be mainly updated to minimize the loss function and satisfy the target relationship in the current triplet, which will lead to an unstable learning process. On the other hand, if α is too small, each iteration will have little influence on the updating of the Mahalanobis matrix. Thus, the metric learning process will be very slow and insufficient. Therefore, the selection of α should consider the tradeoff between efficiency and stability at the same time.

4. Experiments Results

In this section, we conduct experiments on several public domain data sets selected from UCI machine learning repository (<http://archive.ics.uci.edu/ml/>) to present the superiority of the proposed online LDML algorithm and the relationship between the performance and parameters. The parameters of these benchmarks are listed in Table 1. Some of them are with normal size while others have numerous instances or high dimension.

All the following experiments are tested in MATLAB 2011b, and all tests are implemented on a computer with Intel(R) Core(TM) i3-3120 M, 2.50 GHz CPU, 4G RAM, and Windows 7 operating system. The performance index is chosen as the classification accuracy of k -nearest neighbor. The performance of all these algorithms is evaluated using 5-fold cross validation and the final results are the average of results obtained over 5 runs. In our proposed algorithm, when a new instance is received, it will be utilized to randomly build 2 triplets with instances which has been received before. Thus, the total number of the triplets is $2n$. Meanwhile, the learning rate parameter is set as $\alpha = 1/2n$, indicating that each triplet plays the same role in updating the Mahalanobis matrix.

The first experiment aims at illustrating the performance of the proposed compressed representation method in our online LDML algorithm. In this experiment, we try to use various compressed representation with different dimensionality in the metric learning process. The experiments are, respectively, conducted on 3 selected data sets, including "WDBC," "Sonar," and "Ionosphere." The dimensions of these three data sets are 30, 60, and 34. In this test, the number of compressed dimensions varies from 1 to the maximum dimension of the data sets. The cross validation classification precision and the running time which change with the number of dimensions are recorded. And Figure 1 gives the relationship among these three items. We can see that the running time increases exponentially while the precision stays relatively constant when dimensions reach a certain value. The reason for this phenomenon can be explained as follows. Although the Mahalanobis matrix M is used to build the relationship between features and the categories of instances, only a small part of elements in Mahalanobis matrix M make sense. Thus, using a low-rank Mahalanobis matrix F to represent the original M is enough. It is worth noting that if the rank of F is too low, the accuracy will decrease because F does not have enough elements to retain all the important information in metric learning process.

In the second experiment, we compare the proposed method with many other basic classification methods and the state-of-the-art online metric learning algorithms, including Euclidean distance, offline LDML [21], LEGO [13], online ITML [7], and POLA [12]. The experiments are, respectively, conducted on 6 data sets in UCI machine learning repository, including "Iris," "Wine," "WDBC," "Seeds," "Sonar," and "Ionosphere." These data sets are with normal dimension and number of instances. The testing results on cross validation classification accuracy for all data sets are summarized in Table 2. The results list the average and stand deviation of the cross validation classification accuracy over 5 runs. Meanwhile, the number of the compressed dimensions in the proposed online LDML method is also presented in the brackets. From the comparisons we can see, the proposed method outperforms other online metric learning methods as well as Euclidean distance. The precision and robustness of the proposed method are better than all other online metric learning methods. At the same time, compared with the offline LDML, the proposed method only loses a little precision but gains lots of efficiency. Table 3 illustrates

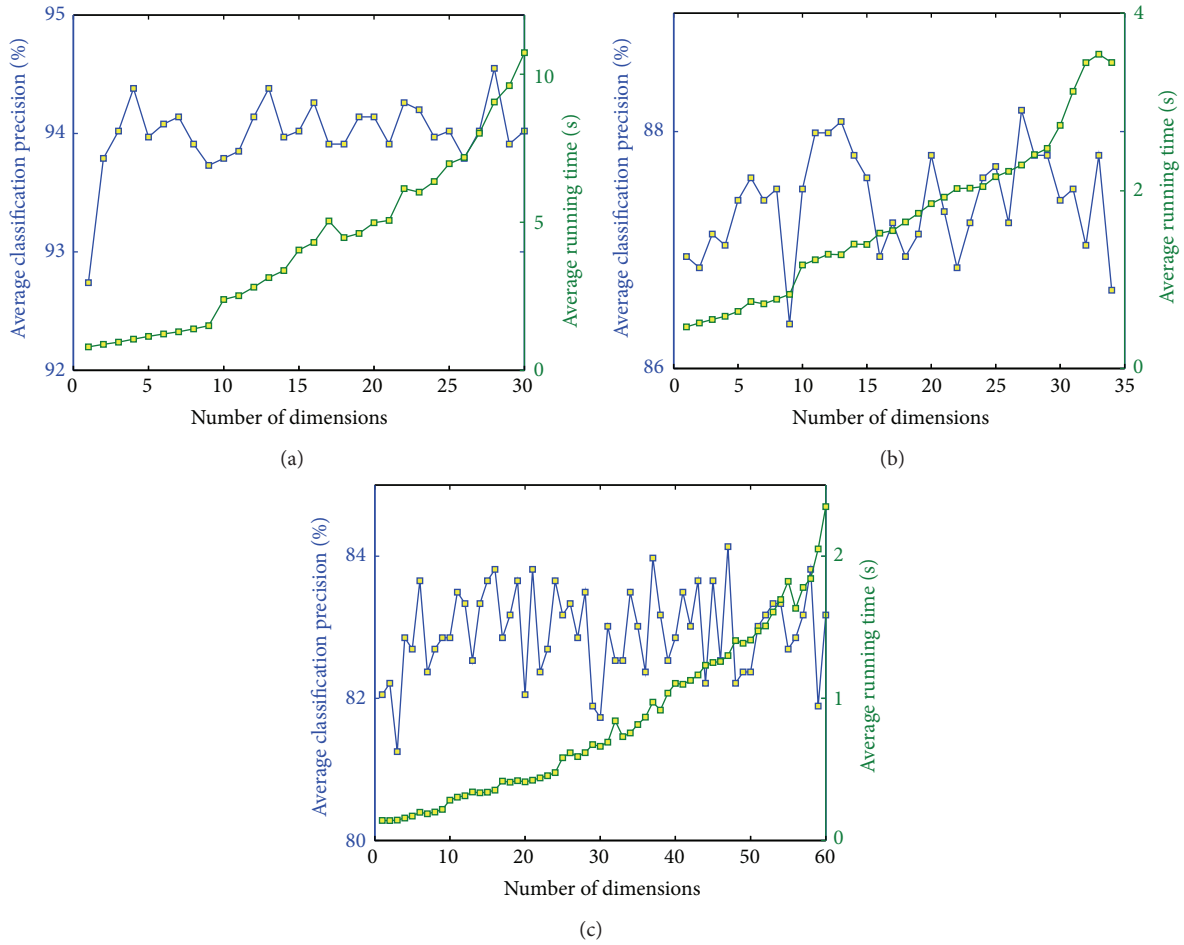


FIGURE 1: The relationship among the number of dimensions, average classification precisions, and average running times. (a) The experiment results on data set “WDBC;” (b) the experiment results on dataset “Ionosphere;” (c) the experiment results on data set “Sonar.”

TABLE 2: Cross validation classification accuracy comparison with the state-of-the-art metric learning methods on the normal size data sets.

Dataset	Offline LDML	Proposed	LEGO	Online ITML	POLA	Euclidean
Iris	0.9800 ± 0.0072	$0.9760 \pm 0.0037(2)$	0.9653 ± 0.0087	0.9507 ± 0.0118	0.9600 ± 0.0166	0.9362 ± 0.0117
Wine	0.9551 ± 0.0221	$0.9461 \pm 0.0152(9)$	0.9132 ± 0.0197	0.7832 ± 0.0228	0.7536 ± 0.0342	0.7427 ± 0.0447
Seeds	0.9543 ± 0.0051	$0.9352 \pm 0.0062(4)$	0.8933 ± 0.0112	0.8760 ± 0.0137	0.8630 ± 0.0152	0.8762 ± 0.0085
WDBC	0.9469 ± 0.0031	$0.9438 \pm 0.0064(4)$	0.9336 ± 0.0029	0.8332 ± 0.0216	0.8822 ± 0.0251	0.8891 ± 0.0202
Sonar	0.8384 ± 0.0120	$0.8279 \pm 0.0023(10)$	0.8250 ± 0.0143	0.8365 ± 0.0186	0.7981 ± 0.0359	0.7240 ± 0.0079
Ionosphere	0.8946 ± 0.0060	$0.8803 \pm 0.0100(13)$	0.8547 ± 0.0122	0.8203 ± 0.0103	0.8131 ± 0.0176	0.8376 ± 0.0191

TABLE 3: Running time (s) comparison with the state-of-the-art metric learning methods on the normal size data sets.

Dataset	Offline LDML	Proposed	LEGO	Online ITML	POLA
Iris	0.4557	0.0306(2)	0.0287	0.0293	0.1372
Wine	0.7019	0.0434(9)	0.0387	0.0399	0.2231
Seeds	0.8957	0.0420(4)	0.0407	0.0420	0.2756
WDBC	3.4414	0.1286(4)	0.1682	0.1620	1.0273
Sonar	1.9713	0.0586(10)	0.1304	0.1291	0.6433
Ionosphere	3.3469	0.1009(13)	0.1229	0.1247	1.0725

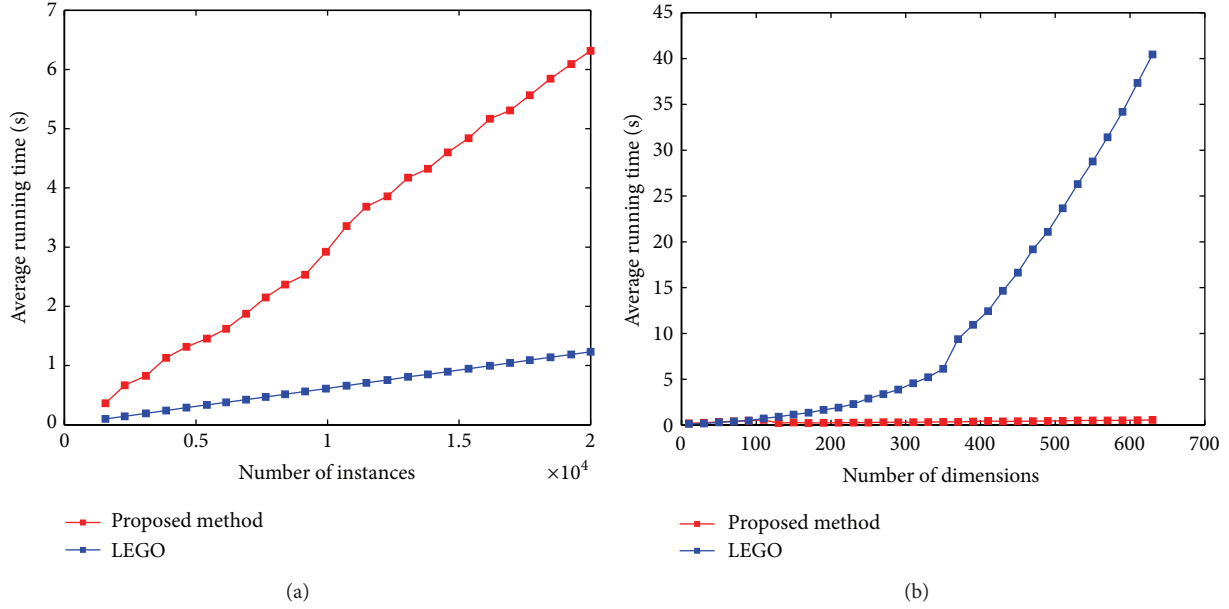


FIGURE 2: The relationship among number of instances, number of dimensions, and average running times. (a) The relationship between number of instances and average running times on data set “Letter-Recognition;” (b) the relationship between number of dimensions and average running times on data set “Mfeat.”

TABLE 4: Performance comparison with the state-of-the-art online metric learning methods on the large data sets.

Data set	LEGO		Proposed algorithm		
	Precision	Time (s)	Compressed dimension	Precision	Time (s)
Letter-Recognition	0.9348	1.2316	10	0.9351	6.3149
Spambase	0.7829	0.7659	20	0.9033	0.9543
Isolet	0.9156	166.9784	20	0.9167	3.2437
Semeion	0.9190	2.4171	20	0.9210	0.1829
Mfeat	0.9375	43.9377	20	0.9496	0.4436

the comparison of running time of these methods. We can see that the LEGO, online ITML, and the proposed method have comparable running times on the normal size data sets. However, our approach has a significant improvement of running time compared with the offline LDML method. Besides, the computational efficiency of the proposed method also outperforms that of POLA a lot.

Then, in the third experiment, tests are conducted on several large data sets to demonstrate the accuracy as well as efficiency of the online LDML algorithm. We select 5 large data sets from UCI machine learning repository, including “Letter-Recognition,” “Spambase,” “Isolet,” “Semeion,” and “Mfeat.” From Table 1 we can see some of them contain numerous instances while others are with high dimensional feature space. We mainly compare the proposed method with LEGO algorithm. The average accuracy and the average running time are illustrated in Table 4. We can see that the precisions of the proposed method on all data sets are better than that of LEGO, especially on the data set “Spambase.” When it comes to the running time, the performance of these two methods is totally different in different data sets. The LEGO algorithm runs fast on “Letter-Recognition” but

it has low efficiency on other data sets, including “Isolet” and “Mfeat.” However, the proposed method can reduce computational complexity on “Isolet” and “Mfeat” but it cannot run fast on “Letter-Recognition.” The reason of these phenomena is not obvious, and further experiments have been conducted to explain these findings.

In the following experiments, we test the relationship among number of instances, number of dimensions, and average running times. In this experiment, we firstly compare the changes of the running time when the number of instances increases. The experiment is conducted on the data set “Letter-recognition” and the result is shown in Figure 2(a). Although the running time of both methods is linear to the number of instances. The running time of the proposed method is a little faster than that of LEGO. The reason is that the online LDML requires computing the orthogonal basis U . Although we have applied the truncated incremental SVD to compute the orthogonal basis, the computational complexity of truncated incremental SVD is $O(nk^3)$. And it can reduce running time sharply only when $k \ll d$. However, in the case of “Letter-Recognition,” the d is 26 and k is 10. The truncated incremental SVD can not work efficiently in this

situation. Another experiment is to illustrate the changes of the running time when the number of dimensions increases. The experiment is conducted on the data set “Mfeat” and the result is shown in Figure 2(b). In this experiment, we gradually increase the feature dimension of the original data while online LDML only uses a 20 dimensional compressed representation all the time. We can see that the running time of proposed method stays of a very low value while that of LEGO increases the square of the number of feature dimensions. Therefore, the LEGO cannot deal with data sets with large feature dimension. The proposed method can reduce lots of computation time in each iteration while keeping high classification performance. This is the main advantage of the online LDML algorithm.

5. Conclusion

In this paper we propose a fast and robust metric learning algorithm for large data sets classification. Since large data sets usually contain numerous instances represented in high dimensional feature spaces, we propose to use an online Logdet divergence based metric learning model to improve the computation efficiency of learning with thousands of instances. Furthermore, we use a compressed representation of high dimensional Mahalanobis matrices to reduce the computational complexity in each iteration significantly. The proposed algorithm is shown to be efficient, robust, and precise by experiments on benchmark data sets and comparison with state-of-the-art algorithms. In future work we plan to further optimize the proposed algorithm with respect to computation efficiency and precision.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

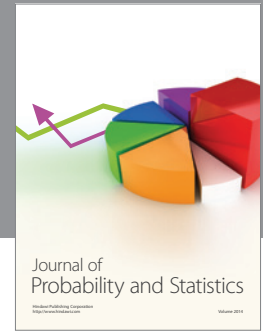
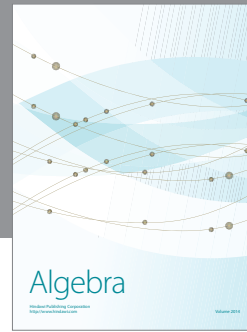
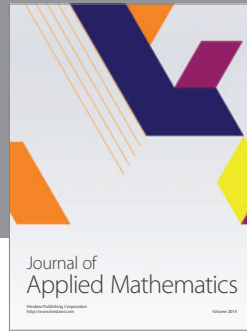
Acknowledgments

The research leading to these results has partially received funding from the Polish-Norwegian Research Programme operated by The National Centre for Research and Development under the Norwegian Financial Mechanism 2009–2014 in the frame of Project Contract no. Pol-Nor/200957/47/2013. Also, this work has been partially funded by the Research Council of Norway (RCN) under Grant 225231/O70.

References

- [1] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, “A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark tennessee eastman process,” *Journal of Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.
- [2] S. Yin, G. Wang, and H. R. Karimi, “Data-driven design of robust fault detection system for wind turbines,” *Mechatronics*, 2013.
- [3] S. Yin, S. X. Ding, A. H. Abandan Sari, and H. Hao, “Data-driven monitoring for stochastic systems and its application on batch process,” *International Journal of Systems Science*, vol. 44, no. 7, pp. 1366–1376, 2013.
- [4] H. Gao, C. Ding, C. Song, and J. Mei, “Automated inspection of e-shaped magnetic core elements using k-tsl-center clustering and active shape models,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1782–1789, 2013.
- [5] M. Guillaumin, J. Verbeek, and C. Schmid, “Is that you? Metric learning approaches for face identification,” in *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV '09)*, pp. 498–505, October 2009.
- [6] E. P. Xing, M. I. Jordan, S. Russell, and A. Ng, “Distance metric learning with application to clustering with side-information,” in *Advances in Neural Information Processing Systems*, pp. 505–512, 2002.
- [7] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, “Information-theoretic metric learning,” in *Proceedings of the ACM 24th International Conference on Machine Learning (ICML '07)*, pp. 209–216, June 2007.
- [8] P. Jain, B. Kulis, J. V. Davis, and I. S. Dhillon, “Metric and kernel learning using a linear transformation,” *Journal of Machine Learning Research*, vol. 13, pp. 519–547, 2012.
- [9] K. Weinberger, J. Blitzer, and L. Saul, “Distance metric learning for large margin nearest neighbor classification,” in *Advances in Neural Information Processing Systems*, vol. 18, p. 1473, 2006.
- [10] S. Yin, H. Luo, and S. Ding, “Real-time implementation of fault-tolerant control systems with performance optimization,” *IEEE Transactions on Industrial Electronics*, vol. 61, no. 5, pp. 2402–2411, 2012.
- [11] J. V. Davis and I. S. Dhillon, “Differential entropic clustering of multivariate gaussians,” in *Advances in Neural Information Processing*, pp. 337–344, 2006.
- [12] S. Yin, X. Yang, and H. R. Karimi, “Data-driven adaptive observer for fault diagnosis,” *Mathematical Problems in Engineering*, vol. 2012, Article ID 832836, 21 pages, 2012.
- [13] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng, “Online and batch learning of pseudo-metrics,” in *Proceedings of the ACM 21st International Conference on Machine Learning (ICML '04)*, p. 94, July 2004.
- [14] P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman, “Online metric learning and fast similarity search,” in *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS '08)*, pp. 761–768, December 2008.
- [15] B. Kulis, M. Sustik, and I. Dhillon, “Learning low-rank kernel matrices,” in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 505–512, June 2006.
- [16] M. Sugiyama, “Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis,” *Journal of Machine Learning Research*, vol. 8, pp. 1027–1061, 2007.
- [17] Q. Wang, P. C. Yuen, and G. Feng, “Semi-supervised metric learning via topology preserving multiple semi-supervised assumptions,” *Pattern Recognition*, vol. 46, no. 9, pp. 2576–2587, 2013.
- [18] J. V. Davis and I. S. Dhillon, “Structured metric learning for high dimensional problems,” in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pp. 195–203, August 2008.

- [19] H. Zha and H. D. Simon, "On updating problems in latent semantic indexing," *SIAM Journal on Scientific Computing*, vol. 21, no. 2, pp. 782–791, 1999.
- [20] H. Zhao, P. C. Yuen, and J. T. Kwok, "A novel incremental principal component analysis and its application for face recognition," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 36, no. 4, pp. 873–886, 2006.
- [21] J. Mei, M. Liu, H. R. Karimi, and H. Gao, "Logdet divergence based metric learning using triplet labels," in *Proceedings of the Workshop on Divergences and Divergence Learning (ICML '13)*, 2013.



Hindawi

Submit your manuscripts at
<http://www.hindawi.com>

