JMLR: Workshop and Conference Proceedings 63:286–301, 2016 ACML 2016

Learning Feature Aware Metric

Han-Jia Ye YEHJ@LAMDA.NJU.EDU.CN
De-Chuan Zhan ZHANDC@LAMDA.NJU.EDU.CN
Xue-Min Si SIXM@LAMDA.NJU.EDU.CN
Yuan Jiang JIANGY@LAMDA.NJU.EDU.CN
National Key Laboratory for Novel Software Technology, Nanjing University,
Nanjing, 210023, China

Editors: Robert J. Durrant and Kee-Eung Kim

Abstract

Distance Metric Learning (Dml) aims to find a distance metric, revealing feature relationship and satisfying restrictions between instances, for distance based classifiers, e.g., kNN. Most Dml methods take all features into consideration while leaving the feature importance identification untouched. Feature selection methods, on the other hand, only focus on feature weights and are seldom directly designed for distance based classifiers. In this paper, we propose a Feature AwaRe Metric learning (Farm) method which not only learns the appropriate metric for distance constraints but also discovers significant features and their relationships. In Farm approach, we treat a distance metric as a combination of feature weighting and feature relationship discovering factors. Therefore, by decoupling the metric into two parts, it facilitates flexible regularizations for feature importance selection as well as feature relationship constructing. Simulations on artificial datasets clearly reveal the comprehensiveness of feature weighting for Farm. Experiments on real datasets validate the improvement of classification performance and the *efficiency* of our Farm approach.

Keywords: Distance Metric Learning; Important Feature Identification; Feature Aware Metric

1. Introduction

The performance of distance based classifiers such as kNN and RBF kernel method mainly depends on the distance between instances, and a well defined distance may lead to high generalization performance. Distance Metric Learning (Dml) aims to find a proper distance which can reveal the true data distribution well and facilitates the classification. Existing researches mainly focus on the Mahalanobis distance (Weinberger and Saul, 2009). Given instances $\{\mathbf{x}_i, \mathbf{x}_j\} \in \mathcal{R}^d$, the (squared) Mahalanobis distance with metric $M \in \mathcal{R}^{d\times d}$ is defined as:

$$\operatorname{dist}^2_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M(\mathbf{x}_i - \mathbf{x}_j)\ . \tag{1}$$

The metric M is a symmetric Positive Semi-Definite matrix, which implies a certain type of relationships between features (Friedman et al., 2008).

Since ordinary Dml methods usually learn metrics with few zero elements, we consider this type of metrics as *full* metric which implicitly indicates the general relationship between all features. There are lots of on-the-shelf Dml methods for learning the *full* metric, e.g., LMNN (Weinberger and Saul, 2009) and ITML (Davis et al., 2007). By neglecting the relationship between features, Dml algorithms can degenerate to feature weighting methods (Gao et al., 2014). In this case, the learned metric is restricted to a *diagonal* matrix whose diagonal elements reflect the importance of each feature.

Full and *diagonal* metrics are two directions of upmost extents. In detail, *full* metric lacks of the ability on distinguishing the importance of features, while *diagonal* metric does not have the ability of figuring the feature correlations out. Structured *sparse* metric is desirable: zero elements in entire rows/columns indicate irrelevancy of corresponding features and the remaining ones reflect the relationship between features. Researchers focus on learning this type of structured *sparse* metric by directly imposing types of sparse constraints on M (Lim et al., 2013) which either leads to the high computational cost or difficult optimization strategy resulting from the symmetric property of M.

In order to incorporate the abilities of feature relationship discovering and feature selection in one Dml approach, we define the Feature AwaRe Metric (Farm) property of a metric, and consequently propose the Farm approach which jointly discovering the feature relationship and selecting important features. In this work, we point out that a structured *sparse* metric can be decomposed into *full* and *diagonal* parts, which respectively models the feature relationship and controls the model complexity. Sparsity on the weighting part consequently makes the feature aware metric with zero-value columns and rows. To the best of our knowledge, we are the first to perform Dml with feature relationship discovering and feature selection simultaneously in the distance calculation. Except for the composition property of feature aware metric, the *sparse* structure leads to *acceleration* of the entire training procedure of Farm as well, and in consequence, Farm can be applied to high-dimensional scenarios.

The main contributions can be summarized as follows:

- A Feature AwaRe Metric learning (Farm) method is proposed to learn structured *sparse* metric which incorporates feature correlation modeling and feature selection in distance based learning scenarios;

- *Sparse* structures of feature aware metric also yield fast metric learning procedures, and extend the proposed Dml approach to high-dimensional cases;

- Experiment results validate effectiveness, efficiency and robustness of the proposed method on different types of data.

The rest of this paper is organized as follows: we first introduce the related work, followed by the proposed Farm method and its implementation details. Experiments are presented after a discussion on the differences between Farm and other sparse metric learning methods. Finally we conclude this work.

2. Related Work

Dml aims to learn a metric for better distance based classification in supervised learning paradigm by utilizing different regularizations and types of side information. For instance, information theoretical approaches based on Bregman optimization (Davis et al., 2007); large margin constraints forced between instances in a triplet (Weinberger and Saul, 2009).

(Kulis, 2012) and (Bellet et al., 2013) provide a concrete review on metric learning algorithms. These Dml methods directly learn metrics based on all features while leaving considerations of feature importance untouched.

Feature importance evaluation is also a prominent issue for generalization abilities, especially for cases with irrelevant features (Friedman et al., 2008) (Choi et al., 2010) (Azmandian et al., 2012). Feature selection methods filter the irrelevance and reduce the computational burden usually by utilizing sparse learning techniques (Hastie et al., 2009), e.g., Lasso (Tibshirani, 1996) and sparse SVM (Bi et al., 2003). In addition, $\ell_{2,1}$-norm for matrices is also often used in multi-task (Liu et al., 2009) and multi-view learning (Wang et al., 2013) to achieve a sparse set of common features. Feature selection/weighting can be regarded as a degradation of learning on *diagonal* metric (Gao et al., 2014), which obviously neglects the interactions between features and will seriously affect the distance calculation.

3. Farm Approach

This section gives the detailed description of Feature AwaRe Metric learning (Farm) approach after a preliminary notation explanation.

3.1. Notations

In supervised Distance Metric Learning (Dml) task, it is supposed there are N instances $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Each instance $\mathbf{x}_i \in \mathcal{R}^d$ with label $y_i \in \{1, \dots, C\}$ and C is the number of classes. $\mathcal{S}_d \subseteq \mathcal{R}^{d\times d}$ is the set of symmetric matrices, and $\mathcal{S}_d^+$ is the Positive Semi-Definite (PSD) cone. The goal of Dml is to learn a metric $M \in \mathcal{S}_d^+$ satisfying the constraints derived from side information. For $\mathbf{w} \in \mathcal{R}^d$, ℓ_1-norm of $\mathbf{w}$, i.e., $\|\mathbf{w}\|_1 = \sum_{i=1}^d |\mathbf{w}_i|$, is the sum of absolute values of each elements. $\mathrm{Tr}(\cdot)$ is the trace of a matrix and $\|M\|_F = \sqrt{\mathrm{Tr}(MM^\top)}$ is the Frobenius norm of matrix M. M's $\ell_{2,1}$-norm is the sum of ℓ_2-norm value of all its rows, while its nuclear norm $\|M\|_*$ is the sum of all its singular values. Operator $[\cdot]_+ = \max(\cdot, 0)$ preserves the positive part of its input value.

3.2. Decoupling of Feature AwaRe Metric

Mahalanobis distance $\mathrm{dist}_M^2(\mathbf{x}_i, \mathbf{x}_j)$ in Eq. 1 is often used to measure similarity between two instances $\mathbf{x}_i$ and $\mathbf{x}_j$. Mahalanobis distance calculation makes use of the full set of features while it is the truth that there are irrelevant and redundant features in concrete applications. Therefore original Mahalanobis distance cannot explicitly indicate the feature importance in the distance calculation. Feature selection, on the other hand, can pick up important feature sets while leaving the high-order feature relationship (e.g., pairwise relationship) seldom considered. In order to calculate distances with *key* features in accordance with the data distribution, it is desired to incorporate a mechanism for identifying helpful features with *sparse* weighting coefficients.

In order to solve the above problem, we introduce a novel Dml approach after defining the property of Feature AwaRe Metric (Farm), which restricts a matrix $\hat{M} \in \mathcal{S}_d^+$ with entire rows and corresponding columns as zero elements. It is obvious that the zero rows and columns are related to features that should be neglected during the distance calculation, while the remaining non-zero elements in $\hat{M}$ model the feature relationship and contribute to the distance value computation.

Directly forcing the properties of feature aware metric leads to complicated optimization problems, such as symmetric $\ell_{2,1}$-norm minimization, which is generally considered as a hard problem (Lim et al., 2013). In this paper, we decouple the feature aware metric $\hat{M}$ into a *full* metric M and a *diagonal* weight separately as follows:

$$\hat{M} = \operatorname{diag}(\mathbf{w}) M \operatorname{diag}(\mathbf{w}) \ , \tag{2}$$

the diag$(\cdot)$ operator transforms a weighting vector $\mathbf{w} \in \mathcal{R}^d$ to a diagonal matrix, i.e., $\operatorname{diag}(\mathbf{w}) \in \mathcal{S}_d$, using *sparse* vector $\mathbf{w}$ as its diagonal elements. The main difference between metric $\hat{M}$ in Eq. 2 and the metric used in Mahalanobis distance in Eq. 1 is that the feature aware metric is considered by combining a *full* metric M with an additional weights vector $\mathbf{w}$. It is obvious that $\mathbf{w}_i = 0$ results in zero-value for i-th row and column, which means the corresponding feature i is useless and the relationships between the i-th feature and others will be neglected as well. From the structure and constraints of $\mathbf{w}$ and M, we can regard them as a feature selector and a relationship constructor, respectively. The decoupling of feature aware metric makes it more flexible than the traditional one, e.g. structured $\hat{M}$ can degenerate to a *full* metric when elements in weights $\mathbf{w}$ all equal to 1, and the structured sparsity of $\hat{M}$ can be preserved by sparse constraint on $\mathbf{w}$. It is noteworthy that $\hat{M}$ can be easily proved as a valid PSD metric. Consequently, distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ using new metric $\hat{M}$ can be summarized as:

$$\operatorname{dist}^2_{\hat{M}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top \operatorname{diag}(\mathbf{w}) M \operatorname{diag}(\mathbf{w}) (\mathbf{x}_i - \mathbf{x}_j) \ . \tag{3}$$

The $\operatorname{dist}^2_{\hat{M}}(\mathbf{x}_i, \mathbf{x}_j)$ can be understood from two aspects: first, $\operatorname{diag}(\mathbf{w}) M \operatorname{diag}(\mathbf{w})$ is the feature aware metric, hence the difference is measured by taking feature importance into consideration; second, $(\mathbf{x}_i - \mathbf{x}_j)^\top \operatorname{diag}(\mathbf{w})$ is a weighted difference between instances, i.e., the feature aware distance metric can be regarded as a Mahalanobis distance with *full* metric M on weighted instance space.

3.3. Flexible Adaptation of Farm Method

By incorporating different regularizers on $\mathbf{w}$ and M, the structure of $\mathbf{w}$ and M can be turned into designed forms and can be used for preserving the structure of $\hat{M}$ as well. In this paper, we use triplets $\mathcal{T} = \{\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t\}_{t=1}^T$ as side information. In each triplet, $\mathbf{x}_j^t$ should be more similar to $\mathbf{x}_i^t$ than $\mathbf{x}_k^t$. A satisfied distance metric should make the distances between dissimilar instances larger than those between similar ones. Therefore, objective function of Farm method is:

$$\min_{M,\mathbf{w}} \sum_{t \in \mathcal{T}} \ell(\operatorname{dist}^2_{\hat{M}}(\mathbf{x}_i^t, \mathbf{x}_k^t) - \operatorname{dist}^2_{\hat{M}}(\mathbf{x}_i^t, \mathbf{x}_j^t)) + \lambda_1 \Omega(M) + \lambda_2 \|\mathbf{w}\|_1 \ , \tag{4}$$

where $\hat{M}$ is the metric defined in Eq. 3. Loss term $\ell(\cdot)$ is a decreasing convex function, e.g., hinge loss, which acts as a surrogate for satisfying triplets constraints among distances. There are two regularizers for controlling and inducing prior knowledge into *full* metric and structured *sparse* weighting vector respectively. Trade-off parameters $\lambda_1, \lambda_2 \geq 0$.

The main property of Farm is that it not only takes high-order relationships between features into consideration, but also keeps the sparse property for feature selection of the learned metric. Structure of feature aware metric is controlled by the regularizers on two components of $\hat{M}$. By controlling the ℓ_1-norm of $\mathbf{w}$, elements in $\mathbf{w}$ are forced to be sparse, and hence zero elements in $\mathbf{w}$ will lead to zero-value rows and corresponding columns of $\hat{M}$, which consequently acts as the role of feature selector. Besides structural sparsity, the regularizer Ω on *full* metric component M is also flexible. For example, when $\Omega(M) = \|M\|_F^2$, it can be used to prevent overfitting; when $\Omega(M) = \|M\|_1$, elements in M are also sparse. Thus, in addition to features selected by $\mathbf{w}$, some feature correlations are also selected (Friedman et al., 2008); when $\Omega(M) = \|M\|_*$, M should be low rank and sparse on principal components is forced, so the combined metric $\hat{M}$ is a sparse and low rank one. Hence its block property can be discovered (Richard et al., 2012) to benefit the learning on different feature attribute sets.

The loss term $\ell(\cdot)$ is used for making distances between instances satisfy triplets constraints as much as possible. In our implementation, $\ell(\cdot)$ is instantiated as the smooth hinge loss (Qian et al., 2015):

$$\ell_s(x) = \begin{cases} 0 & \text{if} \quad x \geq 1 \\ \frac{1}{2} - x & \text{if} \quad x \leq 0 \\ \frac{1}{2}(1-x)^2 & \text{otherwise} \ , \end{cases}$$

which keeps a margin between the dissimilar pairs to improve generalization ability and is convenient for optimization as well. Regularization on *full* metric is configured as $\Omega(M) = \|M\|_F^2$ to make it robust to overfitting.

3.4. Optimization Strategy

Farm method in Eq. 4 jointly optimizes on a *full* metric M and feature weighting vector $\mathbf{w}$. Due to the relevance between optimization variables, we solve in an alternative style, i.e., we optimize on $\mathbf{w}$ with M fixed and vice versa. In each subproblem, it is an optimization problem on a smooth hinge loss plus a regularizer, and both of them can be optimized efficiently. Detailed optimization process is summarized as follows.

Fix w and solve M: When feature weighting vector $\mathbf{w}$ is fixed, it equals to a traditional metric learning problem on weighted instances. We can define the transformed instance: $\hat{\mathbf{x}} = \text{diag}(\mathbf{w})\mathbf{x} = \mathbf{w} \odot \mathbf{x} \in \mathcal{R}^d$, where $\odot$ means the element-wise product. Thus we only need to solve the problem on transformed instances $\hat{\mathbf{x}}$:

$$\begin{aligned} &\min_{M \in \mathcal{S}_d^+} \sum_{t \in \mathcal{T}} \ell_s(\text{dist}_M^2(\hat{\mathbf{x}}_i^t, \hat{\mathbf{x}}_k^t) - \text{dist}_M^2(\hat{\mathbf{x}}_i^t, \hat{\mathbf{x}}_j^t)) + \lambda_1 \|M\|_F^2 \\ = &\min_{M \in \mathcal{S}_d^+} \sum_{t \in \mathcal{T}} \ell_s(\langle M, (\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_k^t)(\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_k^t)^\top - (\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_j^t)(\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_j^t)^\top \rangle + \lambda_1 \|M\|_F^2 \ , \end{aligned} \tag{5}$$

where $\langle A, B \rangle = \text{Tr}(A^\top B)$. For simplicity, we define $\hat{A}^t = (\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_k^t)(\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_k^t)^\top - (\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_j^t)(\hat{\mathbf{x}}_i^t - \hat{\mathbf{x}}_j^t)^\top \in \mathcal{S}_d$. Due to the smooth property of the whole objective, this subproblem can be solved with accelerated projected gradient descent (Li et al., 2014). This method acts in a gradient descent manner and projects current solution to the feasible domain after each descent operation. It is accelerated using Nesterov's method (Nesterov, 2004). The gradient of smooth hinge loss $\ell_s(\cdot)$ w.r.t. *full* metric M can be decomposed to the sum of the gradient of each component $\frac{\partial \sum_{t\in\mathcal{T}} \ell_s(\langle M,\hat{A}^t\rangle)}{\partial M} = \sum_{t\in\mathcal{T}} \frac{\partial \ell_s(\langle M,\hat{A}^t\rangle)}{\partial M}$, and

$$\frac{\partial \ell_s(\langle M,\hat{A}^t\rangle)}{\partial M} = \begin{cases} 0 & \text{if} \quad \langle M,\hat{A}^t\rangle \geq 1 \\ -\hat{A}^t & \text{if} \quad \langle M,\hat{A}^t\rangle \leq 0 \\ (\langle M,\hat{A}^t\rangle - 1)\hat{A}^t & \text{otherwise} \end{cases} .$$

Then we just need to compute the gradient for each triplet and then sum them together to get the gradient of all triplets. The gradient of the subproblem in Eq. 5 can be obtained:

$$\sum_{t\in\mathcal{T}} \frac{\partial \ell_s(\langle M,\hat{A}^t\rangle)}{\partial M} + \lambda_1 M \ .$$

After gradient step, the current solution should be projected back to the PSD cone, which can be done by neglecting eigen-vectors with negative eigen-values after eigen-decomposition. **Fix M and solve $\mathbf{w}$**: When M is fixed, we can do the following transformation on distance:

$$\begin{aligned} \text{dist}^2(\mathbf{x}_i,\mathbf{x}_j) &= (\mathbf{x}_i-\mathbf{x}_j)^\top \text{diag}(\mathbf{w}) M \text{diag}(\mathbf{w})(\mathbf{x}_i-\mathbf{x}_j) \\ &= \mathbf{w}^\top \text{diag}(\mathbf{x}_i-\mathbf{x}_j) M \text{diag}(\mathbf{x}_i-\mathbf{x}_j)\mathbf{w} \ . \end{aligned}$$

Thus distance can be represented as a quadratic form on feature selection weights $\mathbf{w}$. With the transformation, the difference of distances for the t-th triplet can be transformed as:

$$\begin{aligned} \text{dist}^2(\mathbf{x}_i^t,\mathbf{x}_k^t) - \text{dist}^2(\mathbf{x}_i^t,\mathbf{x}_j^t) &= \mathbf{w}^\top \text{diag}(\mathbf{x}_i^t-\mathbf{x}_k^t) M \text{diag}(\mathbf{x}_i^t-\mathbf{x}_k^t)\mathbf{w} \\ &\quad - \mathbf{w}^\top \text{diag}(\mathbf{x}_i^t-\mathbf{x}_j^t) M \text{diag}(\mathbf{x}_i^t-\mathbf{x}_j^t)\mathbf{w} \\ &= \mathbf{w}^\top A^t \mathbf{w} \ . \end{aligned}$$

Here we use $A^t = \text{diag}(\mathbf{x}_i^t-\mathbf{x}_k^t) M \text{diag}(\mathbf{x}_i^t-\mathbf{x}_k^t) - \text{diag}(\mathbf{x}_i^t-\mathbf{x}_j^t) M \text{diag}(\mathbf{x}_i^t-\mathbf{x}_j^t) \in \mathcal{S}_d$ to denote the term not related with $\mathbf{w}$. So the optimization subproblem on $\mathbf{w}$ becomes:

$$\min_{\mathbf{w}} \sum_{t\in\mathcal{T}} \ell_s(\mathbf{w}^\top A^t \mathbf{w}) + \lambda_2 \|\mathbf{w}\|_1 \ . \tag{6}$$

This subproblem is a composite of smooth loss function $\ell_s(\cdot)$ and non-smooth regularizer ℓ_1-norm, and we solve it by fast iterative shrinkage-thresholding algorithm (FISTA) (Beck and Teboulle, 2009). The FISTA method does gradient descent on the smooth part of the objective function in Eq. 6 and then optimizes the non-smooth ℓ_1-norm term. The gradient of smooth hinge loss w.r.t. weighting vector $\mathbf{w}$ can be computed similarly like the gradient of M:

$$\nabla \ell_s(\mathbf{w}) = \frac{\partial \sum_{t\in\mathcal{T}} \ell_s(\mathbf{w}^\top A^t \mathbf{w})}{\partial \mathbf{w}} = \sum_{t\in\mathcal{T}} \frac{\partial \ell_s(\mathbf{w}^\top A^t \mathbf{w})}{\partial \mathbf{w}} \ ,$$

and

$$\frac{\partial \ell_s(\mathbf{w}^\top A^t \mathbf{w})}{\partial \mathbf{w}} = \begin{cases} 0 & \text{if} \quad \mathbf{w}^\top A^t \mathbf{w} \geq 1 \\ -2A^t\mathbf{w} & \text{if} \quad \mathbf{w}^\top A^t \mathbf{w} \leq 0 \\ 2(\mathbf{w}^\top A^t \mathbf{w} - 1)A^t\mathbf{w} & \text{otherwise} \end{cases} .$$

After using gradient descent to get an intermediate solution on current solution $\mathbf{w}'$, FISTA updates $\mathbf{w}$ by solving a proximal sub-problem:

$$\text{prox}_{\lambda_2\|\cdot\|_1}(\mathbf{w}') = \min_{\mathbf{w}} \frac{L}{2}\|\mathbf{w} - (\mathbf{w}' - \frac{1}{L}\nabla\ell_s(\mathbf{w}'))\|_2^2 + \lambda_2\|\mathbf{w}\|_1 \ , \tag{7}$$

where $L > 0$ is the Lipschitz constant of smooth objective, which can be tuned by backtracking strategy. The proximal subproblem in Eq. 7 takes an intermediate solution after gradient step as input which can be solved in closed form (Parikh and Boyd, 2013). It minimizes the non-smooth term $\|\mathbf{w}\|_1$ in the neighborhood of the intermediate solution $\mathbf{w}'$.

Optimization step on M and $\mathbf{w}$ is iteratively conducted. Optimizing M focuses on exploiting the feature relationship for distance calculation, and optimization performed on $\mathbf{w}$ aims to select features given the current *full* metric M. It is notable that as iterations proceed, the sub-problem will focus more on important features. Therefore it is convenient for distance computation due to the redundant feature removing. In each subproblem, our method can guarantee to get an optimal solution, and the objective value can be decreased, so the whole objective function can be decreased in each run. Therefore, the Farm approach will eventually obtain a local optimal solution. The convergence can be proved by the iterative decrease of the function value, and it is also validated in experiments.

4. Implementation Details

Due to the alternative optimization strategy used in our Farm method, the initialization plays a significant role to get good performance on the convergence rate. In our implementation, we solve feature selection weights $\mathbf{w}$ first with an identity matrix to initialize metric M. The reason lies in the fact that: optimization on $\mathbf{w}$ is relatively less costly since $\mathbf{w}$ is a d-dimension vector, smaller than the size of metric M (a $d \times d$ matrix). After we obtain the updated $\mathbf{w}$, irrelevant and some redundant features are removed, therefore the computational burden of the following stage, updating the *full* metric M, can be greatly reduced, since we only need to learn a metric on the selected features. It is noteworthy that the first step of selecting features via updating $\mathbf{w}$ with identity matrix picks up important features as well as preserves distance properties since it equals selecting features in Euclidean space with Dml objectives.

In each iteration of the update on M, only a small portion of M elements are affected, i.e., these elements correspond to relevant features indicated by $\mathbf{w}_i \neq 0$. This can greatly accelerate the M update procedure. It is notable that the initial selection process filters part of features untouched and following updates affect the remaining part of M. M is nearly a *full* metric which is not sparse. In the sub-problem on feature weights $\mathbf{w}$ with fixed metric M, by the relationship $M = LL^\top$, we can use eigen-decomposition to transform last learned metric M to a projection $L \in \mathcal{R}^{d\times d'}$. When computing distances, we only need to decompose M at the start of the sub-problem optimization process and get new projection with $\hat{L} = \text{diag}(\mathbf{w})L$, which equals to elements product for each column of L. With projection $\hat{L}$, the distance can be computed in the Euclidean form in a low dimensional space:

$$\text{dist}^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top \hat{L}\hat{L}^\top(\mathbf{x}_i - \mathbf{x}_j) = \|\hat{L}^\top(\mathbf{x}_i - \mathbf{x}_j)\|^2 \ .$$

The projected dimension also depends on the number of nonzero elements in the current solution of $\mathbf{w}$. Thus, with *sparse* selected features, it can be even smaller than d'.

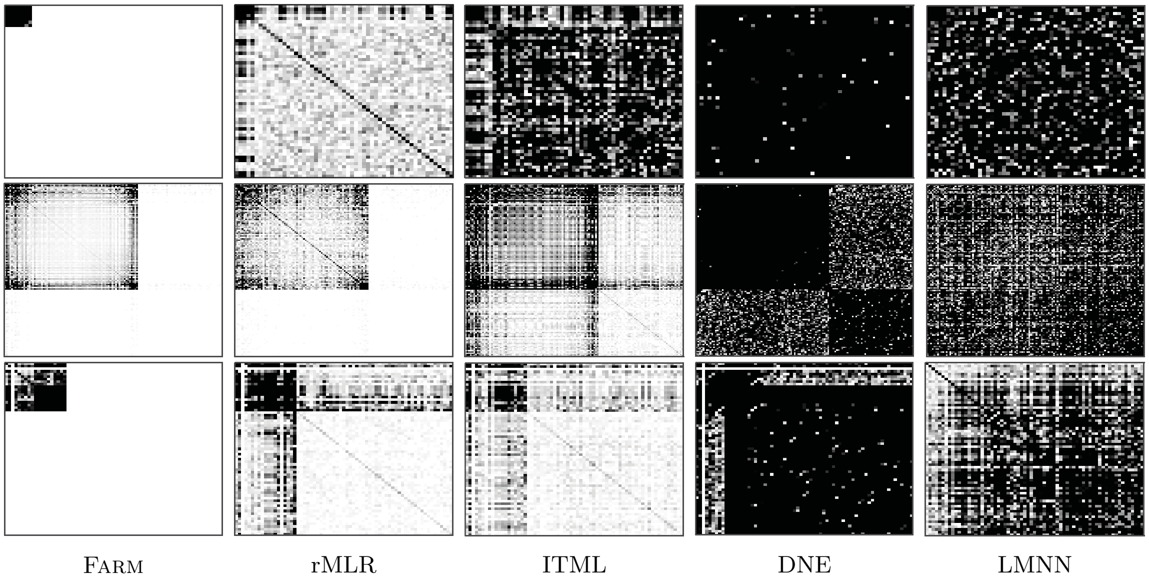


Figure 1: Learned results of different metric learning methods on data with 48-dimension random noise added (each metric is scaled to the same size). The degree of color is in proportion to the absolute value of elements in each metric. Three rows are the results on dataset *autompg*, *mfeat_f*, and *segment* respectively; each column shows results of each method.

5. Discussion

There are some existing researches about learning a *sparse* metric. In (Ying et al., 2009; Huang et al., 2009), the authors consider the Farm properties by learning a metric with sparse principal components. However, they did not directly learn a distance metric in the original feature space, while considering the Farm property in other unitary spaces. This results in their proposed method lacking the ability of feature importance weighting.

And as we have mentioned that, to achieve Farm properties for M, we can directly apply symmetric $\ell_{2,1}$-norm on the *full* metric M. However, direct manipulation of symmetric $\ell_{2,1}$-norm should be with a great challenge. Lim et al. (2013) use both trace norm and symmetric $\ell_{2,1}$-norm to make matrix parameters with jointly row and column sparse property, which can be used in the task of Farm. In their formulation, ADMM technique is involved. However, the entire optimization process is with a heavy burden, because ADMM converges slowly and in each iteration, proximal projections are also costly.

Nevertheless, our Farm approach decouples the Metric $\hat{M}$ into two parts, i.e., a *full* metric M and a structured *sparse* coefficient vector $\mathbf{w}$, and learns $\mathbf{w}$ and M alternatively, which makes the entire Farm approach efficient and can be applied to large scale data. Besides, the decoupled variable $\mathbf{w}$ can directly perform the feature weighting/selection tasks on original feature space where the physical meaning of features is generally preserved. In terms of the usage of our approach, we can perform classification with learned *full* metric in applications where feature relationship is emphasized, or with $\hat{M}$ where both the feature importance and relationship are considered; while for feature selection dominate applications, the $\mathbf{w}$ or $\hat{M}$ can be used in testing as well.

6. Experiments

In experiments, we validate the classification and feature selection performance, scalability and optimization property of our Farm method. In detail, we first show the effectiveness of Farm approach on identifying *key* features on synthetic data, then classification performance of Farm is compared with sate-of-the-art Dml methods as well as feature selection methods. At last, we test the scalability of Farm method and its convergence property.

6.1. Feature Grouping and Feature Importance Detection

The main characteristic of Farm is its ability of identifying *key* features for distance measurement, which has two perspectives. First, as a general feature selection method, features related to labels (classes) should be more important than other features. Second, when concatenated with distance based classifiers, such as kNN, features related to distance computation should have more importance.

We conduct our investigation from synthetic data construction via manipulation of UCI datasets. For dataset $X \in \mathcal{R}^{N\times d}$, we generate a noisy counterpart $Z \in \mathcal{R}^{N\times d_2}$ with $Z_{ij} \sim \mathcal{N}(0,1)$ and obtain the combined data $\hat{X} = [XZ] \in \mathcal{R}^{N\times(d+d_2)}$. The dimension of the noise part equals to 48, i.e., $d_2 = 48$. Farm is compared with rMLR (Lim et al., 2013), ITML (Davis et al., 2007), DNE (Zhang et al., 2007) and LMNN (Weinberger and Saul, 2009). The learned metric of each method is showed in Fig. 1 on datasets *autompg*, *mfeat_f*, and *segment*. Gray levels of pixels are proportioned to the absolute value of corresponding metric coefficients, i.e., important features or feature correlations are with higher gray scale values. Since the original data is placed first in the training input, top left corner of learned metric corresponds to the weights of the original features and hence is expected as the only dark area of the entire plot. From the learned metrics, Farm successfully indicates the area corresponding to noise features on all 3 datasets, i.e., detects the true features related to label. However, DNE, ITML and LMNN almost regard all features contained with noise as useful ones; rMLR can detect true features on the last two datasets, but is still affected by noise from the lower contrast ratio of the entire plots on 3 datasets. From the above experiment, Farm can group features by relevance according to the task, which validates its ability to select features.

To directly test the ability of feature selection of FARM approach, we conduct more investigations with synthetic datasets. We randomly generate 500 instances with 50 dimensions with random seed equal to 100. Each dimension of the data is generated by a normal distribution with different mean and variance values. Thus, there are some minor relationships among different dimensions since they are all created by the same type of distribution but with different parameters. Then we randomly choose 5 dimensions from all features, i.e., [5, 9, 21, 29, 37]. K-Means are applied on all instances but only on the selected dimensions to cluster data into 10 classes and clustering index can be used as labels for instances. Thus, distances used in the clustering are computed only based on the selected features. Consequently, instances of the same class have small distances and the distance measure is only based on the selected 5 features. In addition, after extracting these five dimensions, we check their impact on distance calculation by comparing variations between pairwise distance of all samples and the one computed without one of the 5 dimensions. Average absolute distance derivations are 6.839, 1.375, 0.007, 0.727 and 0.970, which indicate di-

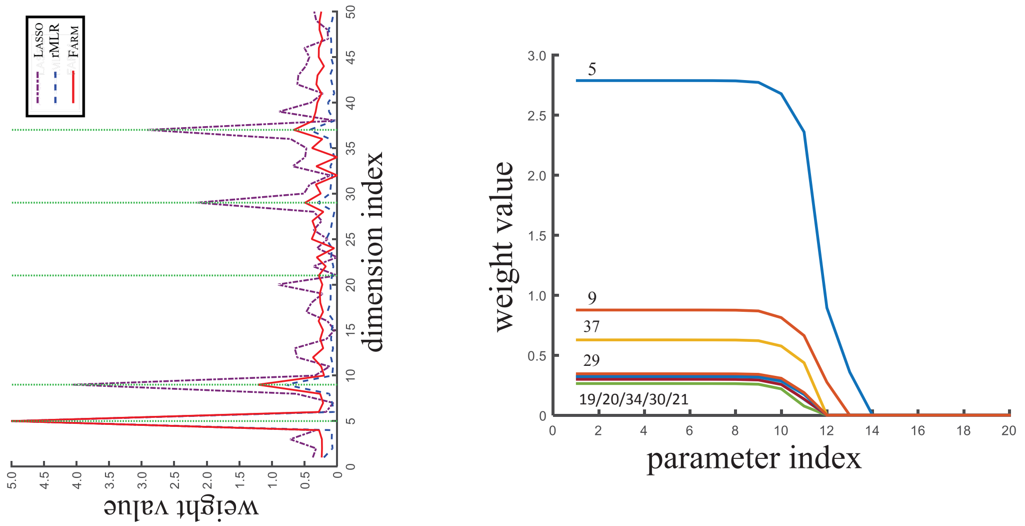


Figure 2: Left: Weight values of Farm, rMLR and Lasso learned on synthetic data. Right: Weighting path of each feature when parameter λ_2 of Farm is changed. For details, please refer to the main context.

mension 5, 9 are the most important ones, and 29, 37 are less important. We can view dimension 21 as irrelevant feature and other unselected dimensions as noise.

Farm is compared with rMLR and Lasso (Liu et al., 2009) on this dataset, which have feature selection ability. rMLR can output a structured metric with sparse rows and columns. In Lasso with square loss, $\ell_{2,1}$-norm is used to make classifier coefficients sparse among different features. The ℓ_2-norm value for each dimension is used as feature weights for both methods' outputs. 70% of the synthetic data are used for training and the remaining is used for test evaluation. The classification errors of Farm, rMLR and Lasso on the test data are 38.01%, 38.60% and 52.63% respectively. Since labels are generated by distance based clustering, Dml can achieve better results than Lasso. From left plot in Fig. 2, feature weights learned by Farm, rMLR and Lasso are in solid line, dash line and dash-dotted line respectively. X-axis is the index for each feature, and y-axis is the normalized weight value. Larger weight value means the corresponding feature is more important. Indexes of selected features used in label generation are annotated by vertical dash-dotted lines. From Fig.2, it is showed that Farm can identify all 4 important latent features and the difference between them is stressed, which is in accordance with their impact on distance value computation. For redundant index 21, Farm gives it less weight compared to the other four but can tell it from noisy dimensions. In contrast, Lasso can find the four key features but will mistakenly regard noises as useful ones. Besides, rMLR can't distinguish differences among learned features while it is able to recognize important ones. The right plot in Fig. 2 shows the weight values of 9 most important features when the sparse parameter λ_2 of Farm is changed. Parameter ranges from 1e-10 to 1e5 with 20 equally log-scale partitions and x-axis shows the index. When parameter becomes smaller, more features will be identified and the more important ones will have larger values. Numbers besides path lines are the indexes of features. From the figure, Farm can detect all 4 important features in accordance

Table 1: Comparisons of classification performance (test errors, mean ± std.) with other methods. Farm$_f$ and Farm$_c$ represent predicts using learned *full* metric M and combined metric respectively. Best results on each dataset are in bold. Last two rows list the win/tie/lose counts on all datasets with t-test against other methods at significance level 95%.

Name	Farm$_f$	Farm$_c$	PLML	LMNN	ITML	EIG	SCML	rMLR	Euclid
autompg	**.204±.035**	.207±.036	.265±.048	.261±.032	.292±.032	.266±.031	.253±.026	.273±.030	.260±.036
clean1	.094±.023	.091±.023	.098±.027	**.084±.020**	.141±.024	.127±.021	.100±.027	.203±.030	.139±.023
fourcla	**.000±.002**	**.000±.001**	**.000±.001**	.001±.001	.002±.003	**.000±.001**	**.000±.000**	.165±.128	.001±.002
german	.278±.019	.281±.022	.280±.016	.291±.020	.288±.021	.284±.014	.302±.021	**.273±.020**	.296±.021
glass	.330±.056	.318±.062	.389±.050	**.301±.046**	.311±.038	.314±.050	.328±.054	.457±.052	.307±.042
hayes-r	**.256±.051**	.260±.046	.436±.201	.305±.065	.342±.080	.289±.067	.296±.053	.335±.056	.398±.046
house-v	**.046±.015**	**.046±.015**	.121±.240	.056±.017	.063±.023	.080±.024	.066±.019	.079±.024	.083±.025
liver-d	.362±.031	.361±.035	.361±.055	**.360±.046**	.377±.052	.380±.037	.371±.042	.372±.049	.384±.040
mfeat_f	.164±.011	**.163±.013**	.183±.021	.171±.009	.189±.010	.229±.079	.185±.012	.181±.012	.201±.010
mfeat_k	.029±.006	.027±.006	.040±.007	**.026±.004**	.039±.007	.051±.008	.047±.008	.036±.007	.044±.007
segment	.030±.007	**.029±.007**	.041±.031	.038±.007	.050±.012	.059±.016	.041±.008	.030±.007	.050±.007
sonar	.147±.050	.150±.039	.171±.048	**.145±.032**	.174±.039	.159±.042	.193±.045	.209±.052	.168±.036
W / T / L	Farm$_f$ vs. others		5 / 7 / 0	6 / 3 / 3	8 / 3 / 1	8 / 4 / 0	8 / 3 / 1	9 / 3 / 0	9 / 1 / 2
W / T / L	Farm$_c$ vs. others		6 / 6 / 0	6 / 5 / 1	8 / 4 / 0	7 / 5 / 0	8 / 3 / 1	9 / 3 / 0	9 / 3 / 0

with their importance in distance computation. Besides, true dimensions used for distance computation are given larger weights compared with other features.

6.2. Comparisons against Dml Methods

To show the effectiveness of the learned metric, we compare Farm approach with some state-of-the-art Dml methods on 12 UCI datasets, namely PLML (Wang et al., 2012), LMNN (Weinberger and Saul, 2009), and ITML (Davis et al., 2007), EIG (Ying and Li, 2012), SCML (Shi et al., 2014), rMLR (Lim et al., 2013). Since Farm learns *full* metric M and combined one $\hat{M}$, we test both of them and denote the results using M and $\hat{M}$ as Farm$_f$ and Farm$_c$ respectively. We do experiments on each data 30 times, and in each trial, we randomly split the whole data into two parts, 70% for training and the remaining for test. Parameters for each training methods are tuned in the first run on the training data, and fixed for all other trials. Each method learns a metric from training data, and the quality of the learned metric is measured with kNN ($k = 3$) on the test data. Mean and standard derivation (std.) of test error are recorded. We also list directly kNN with Euclidean distance as a baseline, which is shown as Euclid in our results Table 1. From comparison results, Farm can achieve best results on 6/12 datasets, and win a lot compared with other methods on most datasets. Due to the page limit, we only report t-test results when Farm$_f$ and Farm$_c$ are compared with DNE, both are 9/3/0. Thus, the learned metric can help improve the ability of subsequent distance based classifier. In addition, this result also shows that the structured *sparse* metric does help for distance computation.

Table 2: Comparisons of classification performance (test errors, mean ± std.) with other methods. Farm$_{\mathbf{w}}$ and Farm$_c$ represent predicts using weights $\mathbf{w}$ and combined metric respectively. Best results on each dataset are in bold. Last two rows list the win/tie/lose counts on all datasets with t-test against other methods at significance level 95%.

Name	Farm$_{\mathbf{w}}$	Farm$_c$	Lasso$_{\text{NN}}$	Lasso	ℓ_1SVM$_{\text{NN}}$	ℓ_1SVM	ℓ_1LR$_{\text{NN}}$	ℓ_1LR	ReliefF
autompg	**.204±.036**	.207±.036	.288±.034	.362±.362	.260±.036	.340±.034	.260±.035	.353±.035	.261±.033
clean1	.124±.022	**.091±.023**	.202±.037	.226±.027	.138±.030	.192±.025	.140±.024	.192±.024	.139±.031
fourcla	**.000±.001**	.001±.002	.333±.026	.307±.021	.333±.026	.313±.022	.333±.026	.313±.020	.329±.028
german	**.278±.018**	.281±.022	.300±.025	.316±.025	.301±.020	.314±.020	.299±.016	.311±.021	.292±.021
glass	**.290±.050**	.322±.065	.327±.046	.455±.034	.307±.042	.421±.041	.306±.042	.434±.040	.300±.049
hayes-r	**.250±.043**	.260±.046	.398±.046	.516±.044	.398±.046	.510±.040	.398±.046	.512±.042	.372±.060
house-v	**.042±.013**	.046±.015	.118±.024	.066±.020	.122±.022	.079±.024	.122±.025	.083±.023	.063±.034
liver-d	.371±.038	**.361±.035**	.398±.045	.383±.042	.400±.041	.372±.041	.400±.041	.375±.039	.386±.050
mfeat_f	**.160±.012**	.163±.013	.184±.011	.220±.012	.201±.010	.246±.013	.201±.010	.251±.012	.177±.010
mfeat_k	.031±.006	**.027±.006**	.042±.006	.068±.008	.044±.007	.086±.009	.044±.007	.086±.009	.034±.006
segment	.030±.006	**.029±.007**	.040±.006	.187±.010	.050±.007	.176±.010	.050±.007	.182±.010	.045±.007
sonar	.163±.041	**.150±.039**	.295±.056	.257±.047	.202±.053	.253±.042	.276±.040	.255±.046	.184±.051
W / T / L	Farm$_{\mathbf{w}}$ vs. others		12 / 0 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	9 / 3 / 0
W / T / L	Farm$_c$ vs. others		11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0	11 / 1 / 0

6.3. Comparisons against Feature Selection Methods

In this part, we will validate the feature selection ability of Farm. We compare our method with feature selection based methods on the above datasets. Besides testing using learned metric $\hat{M}$, we can also select features using the learned weight $\mathbf{w}$ first, and then use kNN to do classification. This type of learner is denoted as Farm$_{\mathbf{w}}$. We compare Farm with some feature selection methods such as those in (Hastie et al., 2009) and (Liu et al., 2009), which use ℓ_1-norm and $\ell_{2,1}$-norm to select important features. These methods denoted as Lasso in the result table. We also make a comparison between our method and classic ReliefF (Robnik-Šikonja and Kononenko, 1997) method, which finds the near-hit and near-miss instances using the Manhattan norm. In addition, ℓ_1-norm regularized SVM and logistic Regression (LR) are also compared. Besides classifying directly using learned weights, we also list the results of kNN on selected features (The name of the feature selection method has a subscript 'NN'). We use the same partition strategy as in sub-section 6.2. Error rates on test data are listed in Table 2, and the best results on each dataset are in bold. From Table 2, it is showed that our Farm method can work better than other feature selection methods. Thus, Farm can select useful features for computing distance and it will help subsequent classification to get better results.

6.4. Investigations on Large Scale Datasets

According to the accelerated implementation of the proposed method, Farm can select more and more *key* features during the training process, and the later optimization and update process will focus more on these selected features for distance computation, thus the training time is greatly reduced.

To test the scalability and efficiency, we run Farm approach on some datasets with large number of instances and especially with high dimensions. The M^{2}LMNN is a multiple

Table 3: Comparisons of classification test errors with other methods. Farm_f, $\text{Farm}_{\mathbf{w}}$ and Farm_c represent predicts using learned *full* metric M, weights $\mathbf{w}$ and their combined metric respectively. Minimum test error on each dataset is in bold.

Name	N	D	Farm_f	$\text{Farm}_{\mathbf{w}}$	Farm_c	PLML	M^2LMNN	LMNN	ITML	EIG	SCML	rMLR	Euclid
Aloi	108000	128	0.050	0.059	**0.049**	-	-	0.053	0.110	-	0.107	-	0.073
Caltech30	5478	500	0.645	0.687	0.644	-	0.589	**0.581**	0.605	0.642	0.723	0.582	0.908
COIL20	1440	1024	**0.000**	0.007	**0.000**	0.016	0.005	0.005	0.009	0.037	0.030	0.076	0.030
Msrcorid_1	4313	1024	0.141	0.166	0.122	-	0.131	**0.114**	0.169	0.165	0.169	0.988	0.168
Msrcorid_2	4313	1536	0.226	0.286	0.236	-	**0.208**	0.230	0.347	0.342	0.393	0.988	0.294
Msrcorid_3	4313	1250	0.193	0.257	**0.188**	-	0.247	0.233	0.246	0.251	0.419	0.226	0.277
Optdigits	5620	65	0.008	0.012	**0.007**	**0.007**	0.008	0.009	0.020	0.012	0.015	0.011	0.018
Orig	70000	784	**0.021**	0.028	**0.021**	-	0.034	0.051	0.054	0.060	0.085	-	0.056
Reut8	7670	500	0.072	0.103	0.076	-	0.066	**0.062**	0.085	0.131	0.239	0.500	0.231
Spambase	4601	58	0.072	0.070	**0.063**	0.072	0.064	0.070	0.085	0.067	0.070	0.096	0.085
UIUC_1	5499	1024	**0.030**	0.050	0.035	-	0.032	0.033	0.063	0.082	0.091	0.872	0.064
UIUC_2	5499	1536	0.148	0.220	**0.143**	-	0.212	0.164	0.246	0.308	-	0.872	0.258
UIUC_3	5499	1250	0.120	0.220	**0.112**	-	0.167	0.161	0.181	0.580	0.724	0.872	0.265
USPS	9298	256	0.025	0.026	**0.024**	-	0.028	0.024	0.035	0.033	0.355	0.050	0.033
VOC2009_1	5254	800	**0.338**	0.424	0.346	-	0.425	0.404	0.402	0.373	0.607	0.414	0.405
VOC2009_2	5254	1536	**0.376**	0.446	0.380	-	-	0.385	0.432	0.462	0.656	0.645	0.438
VOC2009_3	5254	1250	**0.344**	0.416	0.344	-	-	0.395	0.403	0.385	0.591	0.471	0.408
Waveform	5000	40	**0.159**	0.189	0.163	0.167	0.205	0.226	0.238	0.170	0.224	0.236	0.251

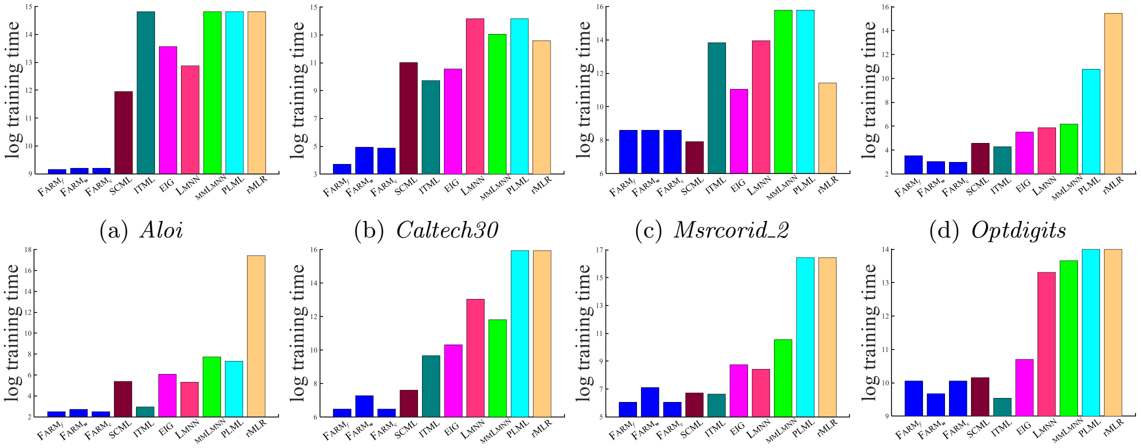
metric learner (Weinberger and Saul, 2009). On each data, we randomly split the data into 3 parts, 40% for training, 30% as validation set and the remaining for test. Each comparison method is tuned on the validation set and then tested using the selected best parameters. Test errors on each data are listed in Table 3. The basic information of each dataset is also listed in the table. N and D denote the number of instances and dimension of each dataset respectively. From these datasets, *Caltech* (Fei-Fei et al., 2007) is object classification dataset, and we use the 30 most frequent classes for test. On image datasets *Msrcorid* [1], *UIUC*[2] and *VOC2009*[3], we extract bag of words, fisher vector and SPM features. We refer these three kinds of feature as index 1, 2, 3 respectively in Table 3.

Our experiments are performed on a cluster of 32 machines, each of which has four 6-core 2.53GHz CPUs and 48G RAM. From the results in Table 3, our Farm method can achieve best results on 14/18 datasets. The '-' notation in the table means the compared method cannot get a result in 24 hours. In addition, the training time of each method on each dataset is also recorded, and 8 of them are showed in Fig. 3. It should be noted that Farm_f, $\text{Farm}_{\mathbf{w}}$ and Farm_c are trained with different parameters, thus they cost different time for training. From Fig. 3, our Farm method can be trained very fast on datasets that have large number of instances and high dimensionality, which validates its efficiency.

[1] http://research.microsoft.com/en-us/downloads/b94de342-60dc-45d0-830b-9f6eff91b301/

[2] https://cogcomp.cs.illinois.edu/Data/Car/

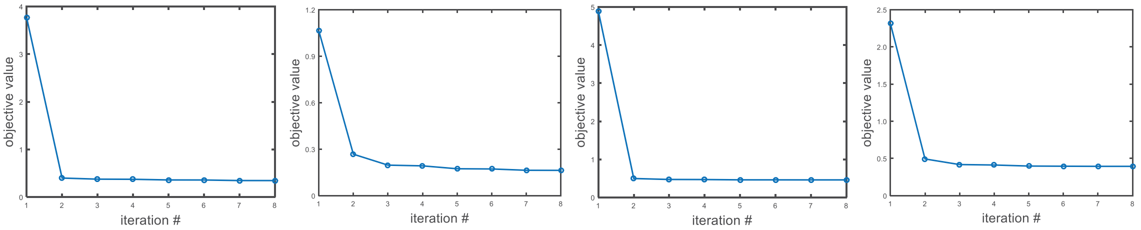
[3] http://host.robots.ox.ac.uk/pascal/VOC/voc2009/index.html



(a) *Aloi* (b) *Caltech30* (c) *Msrcorid_2* (d) *Optdigits*

(e) *Pendigits* (f) *Reut8* (g) *USPS* (h) *VOC2009_2*

Figure 3: Time comparison(in $\log_2$ scale) between Farm and other metric learning methods.



(a) *autompg* (b) *fourclass* (c) *glass* (d) *hayes-roth*

Figure 4: Changes of Farm objective value. X-axis is the number of iteration in our algorithm. In each iteration, $\mathbf{w}$ or M will be updated. Y-axis is the value of the whole objective for Farm.

6.5. Convergence: An Empirical Analysis

Farm is solved in iterative optimization, i.e., alternatively optimizing the feature selection weights $\mathbf{w}$ and *full* metric M in trails. The whole objective in Eq. 4 will be decreased once $\mathbf{w}$ or M is updated. Due to the non-negative property of the whole loss function and the objective decrease property of the solver for each sub-problem, the algorithm should be converged. In order to analyze the converge performance of Farm, we record the change of the objective function on the first run for each datasets. Four of them are listed in Fig. 4. In each figure, the x-axis is the number of sub-iteration, i.e., we record the objective value change once one variable ($\mathbf{w}$ or M) is updated. From the variation plots of objective value, we can summarize two phenomena: first, the objective value will be decreased once a variable is updated and it will be converged at last. This property facilitates the stop criteria of the training process; second, the convergence rate of our algorithm is very fast. On most datasets, it can be converged in less than 3 updates for both $\mathbf{w}$ and M. So in practical implementation, we can get satisfied results in short time.

7. Conclusion

Distance metric learning mainly focuses on learning an appropriate distance measurement with feature relationships considered during distance calculation. Most existing distance metric learning methods neglect to explicitly count on the feature importance. In this work, a Feature AwaRe Metric learning (Farm) approach is proposed, which considers the feature relationships as well as the feature importance via decoupling the metric to be learned into a *full* metric and a *sparse* weights vector. One obvious advantage of metric decoupling lies in the fact of flexibilities on regularizer designed for *full* metric as well as weighting/selection on features with sparse ℓ_1-norm penalty. It also figures out that the Farm approach can be solved efficiently and it is scalable for large datasets due to the separation of *full* metric and *sparse* weights vector. Empirical investigations against distance metric learners and feature selection methods clearly indicate the ability of feature importance detection and distance measuring with high-order feature correlation of Farm approach. Real datasets assessments also validate the effectiveness and efficiency of Farm on classification tasks.

Acknowledgments

The authors would like to thank reviewers for helpful comments. This work was supported by NSFC (61273301) and JiangsuSF BK20131278.

References

Fatemeh Azmandian, Jennifer G Dy, Javed A Aslam, and David R Kaeli. Local kernel density ratio-based feature selection for outlier detection. In *Proc. of 4th ACML*, pages 49–64, 2012.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

Jinbo Bi, Kristin Bennett, Mark Embrechts, Curt Breneman, and Minghu Song. Dimensionality reduction via sparse support vector machines. *JMLR*, 3:1229–1243, 2003.

Nam Hee Choi, William Li, and Ji Zhu. Variable selection with the strong heredity constraint and its oracle property. *Journal of the American Statistical Association*, 105(489):354–364, 2010.

Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th ICML*, pages 209–216, Corvalis, OR., 2007.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Xingyu Gao, Steven CH Hoi, Yongdong Zhang, Ji Wan, and Jintao Li. Soml: Sparse online metric learning with application to image retrieval. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 1206–1212, Quebec, Canada, 2014.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer-Verlag New York, New York, NY., 2009.

Kaizhu Huang, Yiming Ying, and Colin Campbell. GSML: A unified framework for sparse metric learning. In *9th IEEE International Conference on Data Mining, 2009.*, pages 189–198, 2009.

Brian Kulis. Metric learning: A survey. *Foundations and Trends in ML*, 5(4):287–364, 2012.

Nan Li, Rong Jin, and Zhi-Hua Zhou. Top rank optimization in linear time. In *Advances in Neural Information Processing Systems 27*, pages 1502–1510. Cambridge, MA.: MIT Press, 2014.

Daryl Lim, Gert Lanckriet, and Brian McFee. Robust structural metric learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 615–623, Atlanta, GA., 2013.

Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proc. of the 25th Conference on Uncertainty in AI*, pages 339–348, 2009.

Yurii Nesterov. *Introductory lectures on convex optimization*, volume 87. Springer Science & Business Media, Secaucus, NJ., 2004.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1 (3):123–231, 2013.

Qi Qian, Rong Jin, Shenghuo Zhu, and Yuanqing Lin. Fine-grained visual categorization via multi-stage metric learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3716–3724, Boston, MA., 2015.

Emile Richard, Pierre-andre Savalle, and Nicolas Vayatis. Estimation of simultaneously sparse and low rank matrices. In *Proceedings of the 29th ICML*, pages 1351–1358, Edinburgh, Scotland, 2012.

Marko Robnik-Šikonja and Igor Kononenko. An adaptation of relief for attribute estimation in regression. In *Proceedings of the 14th ICML*, pages 296–304, Nashville, TN., 1997.

Yuan Shi, Aurélien Bellet, and Fei Sha. Sparse compositional metric learning. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 2078–2084, Quebec, Canada, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Hua Wang, Feiping Nie, and Heng Huang. Multi-view clustering and feature learning via structured sparsity. In *Proceedings of the 30th ICML*, pages 352–360, Atlanta, GA., 2013.

Jun Wang, Alexandros Kalousis, and Adam Woznica. Parametric local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems 25*, pages 1601–1609. Cambridge, MA.: MIT Press, 2012.

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009.

Yiming Ying and Peng Li. Distance metric learning with eigenvalue optimization. *JMLR*, 13(1): 1–26, 2012.

Yiming Ying, Kaizhu Huang, and Colin Campbell. Sparse metric learning via smooth optimization. In *Advances in Neural Information Processing Systems 22*, pages 2214–2222. 2009.

Wei Zhang, Xiangyang Xue, Zichen Sun, Yue-Fei Guo, and Hong Lu. Optimal dimensionality of metric space for classification. In *Proceedings of the 24th ICML*, pages 1135–1142, 2007.