Multi-feature Metric Learning with Knowledge Transfer among Semantics and Social Tagging

Shuhui Wang[1] Shuqiang Jiang[1] Qingming Huang[1,2] Qi Tian[3]

[1]Key Lab of Intell. Info. Process.(CAS), Inst. of Comput. Tech., CAS, Beijing, 100190, China
[2]Graduate University, Chinese Academy of Sciences, Beijing, 100049, China
[3]Dept. of Computer Science, Univ. of Texas at San Antonio, TX78249, U.S.A.
{shwang, sqjiang, qmhuang}@jdl.ac.cn, qitian@cs.utsa.edu

Abstract

Previous metric learning approaches learn a unified metric for all the classes on single feature representation, thus cannot be directly transplanted to applications involving multiple features, hundreds to thousands of hierarchical structured semantics and abundant social tagging. In this paper, we propose a novel multi-task multi-feature metric learning method which models the information sharing mechanism among different learning tasks. We decompose the real world multi-class problems such as semantic categorization or automatic tagging into a set of tasks where each task corresponds to several classes with strong visual correlation. We conduct metric learning to learn a set of (*hyper*)*category-specific metrics for all the tasks. By encouraging model sharing among tasks, more generalization power is acquired. Another advantage is the capability of simultaneous learning with semantic information and social tagging based on the multi-task learning framework, and thus they both benefit from the information provided by each other. Experiments demonstrate the advantages on applications including semantic categorization and automatic tagging compared with other popular metric learning approaches.*

1. Introduction

The core problem of many image applications is to learn a good metric that can well represent the semantic inter-relationship as the metric usually serves as the *information bottleneck*. For example, for kernel based models [21, 31], good metric usually leads to more robust model and sparser support vectors. For lazy methods [33], a more semantically consistent metric is required so that the retrieved neighborhood contain more samples with consistent class labels. To this end, Distance Metric Learning (DML) [5, 12, 14, 15, 23, 25, 33] formulate the problem with side information using criteria such as *max-margin* [33] and leave-one-out *k*-NN classification error [15]. Although remarkable successes have been achieved on many small scale problems, they cannot be directly transplanted to large scale visual applications. We study the metric learning by addressing the following disadvantages of previous approaches.

Firstly, the images are usually represented by a set of different and complementary features, such as color, texture and Bag-of-Word features. Unfortunately, how to learn a similarity measure with multiple features have rarely been discussed in DML except for simple feature concatenation. The most realistic drawbacks of feature concatenation are risk of over-fitting and computation complexity growth in $O(M^2d^2)$ scale, where M and d represent the number of features and their average number of dimension. Moreover, it is not flexible to process nonlinear kernels since the covariance of different features is hard to compute in nonlinear feature space. To overcome these drawbacks, inspired by Multiple Kernel Learning (MKL) [2, 26] and other metric optimization studies [21, 24, 25, 27, 28], we propose to learn the *Mahalanobis* matrices for each feature channel and their combining coefficients in a unified learning framework. Our model possesses only $O(Md^2)$ computational complexity and capability of learning on nonlinear kernels.

The second challenge for real world applications is that they usually involve more than hundreds of semantic classes with hierarchical structure such as *WordNet* [6, 7]. This brings about extremely cluttered distribution in feature space. If we explore the semantic structure, we find that images from the same semantic subset usually share some visual properties [18, 19, 30], while images from different subsets may be very easy to be classified. For example, *palm tree* and *coconut tree* may be hard to distinguish since their leaves and trunks are very similar, while *flower* and *car* are obviously different in shape and color. Such prior knowledge is beneficial to provide guidance on the construction of information sharing structure among semantic concepts, so that model capacity can be enhanced for learning models towards real world application. To this end, several metric learning approaches have been proposed. For example, Parameswaran *et al*. [28] extended the well performed *LMNN* [33] to multi-task metric learning, and Hwang *et al*. [18] proposed to learn a tree of metrics to incorporate the object hierarchy. Our method is an extension of [28] as we propose a more flexible

978-1-4673-1228-8/12/$31.00 ©2012 IEEE

framework where the tasks can be grouped with different levels of semantic class generalization, so better tradeoff between performance and efficiency can be achieved for large scale applications. Moreover, similar in the spirit of sparsity disjoint regularization [18], we provide an automatic feature weighting to learn feature weights for the "sharing" part and "discriminating" part of the metrics.

The third challenge for Web image analysis is how to develop an effective method for modeling the inter-relation between semantic labels and abundant social tagging in order to collaboratively promote the image understanding. Generally speaking, the semantic information is usually compact, predefined, well labeled but expensive to obtain, while social tagging is more ubiquitous, diversified but contains certain level of noise and personal inclination. Different from previous studies either treating social tagging as a textual feature or assuming that the social tagging contains over-complete semantic information, we also treat the learning with social tagging as a set of tasks as the tasks of learning with semantic concepts. The metrics for visual categorization and automatic tagging are learned jointly based on our multi-task learning framework. This strategy is not only capable of introducing the abundant information contained in social tagging to enhance the learning of semantics, but also capable of reducing noise for image tagging by using clean semantic information.

In this paper, to better model the intrinsic nature of multiple features and the information sharing among different tasks, and to reach better tradeoff between efficiency and accuracy, we propose a multi-task multi-feature similarity learning method which learns the similarities for multiple tasks simultaneously with knowledge propagation among all the tasks. For intra-task similarity learning, it explores the relation of feature sharing and feature weighting among the visually similar semantic groups. For inter-task information propagation, it encourages knowledge transfer from other similarity learning tasks. Compared with other binary multi-task models [11], our method is more economical since the number of learned metrics will grow sub-linearly as the learning strategy is able to process several semantic concepts or tags in a single task. As another technical contribution, we develop an efficient solution for the convex dual problem.

In Section 2 we provide a brief review on relevant studies. We introduce our method in Section 3. Experiments are conducted and discussed in Section 4. In Section 5 we summarize the paper.

2. Related works

Our work is closely related with Multiple Kernel Learning, metric learning, and multi-task learning. We brief some representative works on these studies.

Recently, Multiple Kernel Learning [2, 26] which makes use of multiple features based on multiple kernels achieves great success in computer vision [13, 31]. It was later developed with non-sparse l_p-MKL [32]. The idea of MKL also inspired other research in machine learning, such as dimension reduction [21, 27] and transfer learning [8, 9]. Kundu *et al.* [24] proposed to simultaneously maximize the margin of the model and learn a similarity on multiple similarity matrices by optimizing each single similarity and their kernel combination coefficients together. Our work is inspired by multiple kernel dimensional reduction [21, 27] and [24]. Our formulation is similar with l_p-MKL [32], as non-sparse kernel weight coefficient shows better discriminate power than sparse weight coefficient.

Metric learning has been a canonical problem in pattern recognition and image analysis. The approaches in [33] proposed to formulate DML by applying the large margin principle on neighborhood. The methods proposed by [14, 15], Davis *et al.* [5] and Sugiyama [29] are based on different local loss terms on training data. The kernel learning proposed by Kulis *et al.* [23] is closely related to [5] since distance and similarity (kernel) can be inter-converted. Besides, Kwok *et al.* [25] proposed a kernel learning method based on the *idealized kernel*. Frome *et al.* [12] learned an asymmetric localized similarity for image retrieval and classification. Our work is also inspired by [25] in problem formulation, as we try to simultaneously learn the idealized similarity measures for multiple tasks based on multiple feature representations.

Multi-task learning [3, 11] is a good way to improve the model generalization ability by sharing information [3], features [34, 30, 1] and training samples [11] among different tasks. For visual analysis, multi-task learning is a natural choice because image categories usually share some common features such as corner-like patches [30]. In recent studies, the sharing relation among features across different tasks is usually modeled by group sparsity [1, 19, 34]. Also, Kang *et al.* [20] provided some guidance on task grouping by study the sharing mechanism in multi-task learning. Our work is inspired by two studies [19, 28]. Hwang *et al.* [19] proposed to learn the class labels and visual attributes using multi-task MCSVM, which is similar with semantic labels and social tagging in our study. Parameswaran *et al.* [28] adapted *LMNN* [33] into multi-task DML, which is more flexible than binary models for multi-class problems. However, it is only capable of linear metric learning. For multiple features with thousands of dimensions, the computational cost for [28] is prohibitive.

3. Approach

Given a set of tasks {$\mathbf{T}_t$, t=1,…,T}, and their training data $\mathbf{X}_t = \{(\mathbf{x}_t^i, c_t^i) \mid i = 1,...,N_t\}$, where $\mathbf{x}_t^i$ represents a data of t^{th} task and c_t^i denotes its class label. For each $\mathbf{x}_t^i$, we calculate M types of features, denoted by $x_t^{i,m}, m = 1,...,M$.

For each pair $(\mathbf{x}_t^i, \mathbf{x}_t^j)$ in t^{th} task, the metric are defined by a combination of the "sharing" part and "discriminating" part:

$$\begin{aligned}&\widetilde{K}_t^{ij} = \sum_{m=1}^{M} \widetilde{K}_t^{ij,m},\ \widetilde{K}_t^{ij,m} = (x_t^{i,m})^* \left(A_0^{(m)} + A_t^{(m)} \right) x_t^{j,m} \\ &\widetilde{d}_t^{ij} = \sum_{m=1}^{M} \widetilde{d}_t^{ij,m},\ \widetilde{d}_t^{ij,m} = (x_t^{i,m} - x_t^{j,m})^* \left(A_0^{(m)} + A_t^{(m)} \right)(x_t^{i,m} - x_t^{j,m})\end{aligned} \tag{1}$$

where $(\bullet)^*$ denotes the transpose of matrix. $A_t^{(m)}$ is the *Mahalanobis* matrix of the m^{th} feature channel for t^{th} task, and $A_0^{(m)}$ denotes the *Mahalanobis* matrix for task 0 that is shared by all tasks.

3.1. Model

By incorporating the formulation of *idealized kernel* learning [25], the joint objective function is formulated as:

$$\begin{aligned}\min_{\mathbf{b},\mathbf{A}}\ &\frac{1}{2}\left(\gamma_0 \sum_{m=1}^{M} \frac{1}{b_0^{(m)}} \| A_0^{(m)} \|_F^2 + \sum_{t=1}^{T}\sum_{m=1}^{M} \frac{\gamma_t}{b_t^{(m)}} \| A_t^{(m)} \|_F^2 \right) \\ &+ \frac{C}{N} \sum_{t=1}^{T} \sum_{ij \in S} \xi_t^{ij} + \frac{\eta}{2} \sum_{t=0}^{T} \| \mathbf{b}_t \|_p^2 \\ s.t.\ \ &\delta_t^{ij}\left(d_t^{ij} - \widetilde{d}_t^{ij} \right) \geq \sigma_t^{ij} - \xi_t^{ij},\ \xi_t^{ij} \geq 0, b_t^{(m)} \geq 0, p > 1,\ A_t^{(m)} \succeq 0\end{aligned} \tag{2}$$

where the first two terms regularize the complexity of the learned metric. The third term measures the hinge loss on training data pairs. We denote $\hat{x}_t^{ij,m} = x_t^{i,m} - x_t^{j,m}$, and S represents the labeled pair set with N pairs. C denotes the penalty on training data. δ_t^{ij} denotes the labels of similar/dissimilar labeled pairs, and σ_t^{ij} is a predefined threshold for hinge loss which determines the sparsity of support vectors. The last term is regularization on the kernel weights, where $\mathbf{b}_0$ and $\mathbf{b}_t$ represent weights for the sharing part and discriminating parts. $\| \bullet \|_F$ and $\| \bullet \|_p$ represents the *Frobenius* norm and l_p-norm, respectively. Contrast with sparse kernel combination [2, 26, 31], we adopt p-norm to encourage that the learned metric incorporate controllable non-sparse prior. d_t^{ij} denotes the original metric. The learned *Mahalanobis* matrices are represented by:

$$\begin{aligned}&A_t^{(m)} = -\frac{b_t^{(m)}}{\gamma_t} \sum_{ij \in \mathbf{T}_t} \alpha_t^{ij} \delta_t^{ij} \hat{x}_t^{ij,m}, b_t^{(m)} = \frac{1}{2\gamma_t \eta} (o_t^m)^{q-1} \left(\sum_{m'=1}^{M} \left(o_t^{m'} \right)^q \right)^{\frac{2}{q}-1} \\ &A_0^{(m)} = -\frac{b_0^{(m)}}{\gamma_0} \sum_{t} \sum_{ij \in \mathbf{T}_t} \alpha_t^{ij} \delta_t^{ij} \hat{x}_t^{ij,m}, b_0^{(m)} = \frac{1}{2\gamma_0 \eta} (o_0^m)^{q-1} \left(\sum_{m'=1}^{M} \left(o_0^{m'} \right)^q \right)^{\frac{2}{q}-1} \\ &o_t^m = \boldsymbol{\alpha}_t^* \mathbf{Q}_{t,t}^{(m)} \boldsymbol{\alpha}_t, o_0^{(m)} = \boldsymbol{\alpha}^* \mathbf{Q}^{(m)} \boldsymbol{\alpha}\end{aligned} \tag{3}$$

The dual problem becomes a q-norm convex problem [32]:

$$\begin{aligned}&\min_{\boldsymbol{\alpha}} \sum_{t=1}^{T} \frac{1}{8\gamma_t^2 \eta} \left(\sum_{m=1}^{M} \left(\boldsymbol{\alpha}_t^* \mathbf{Q}_{t,t}^{(m)} \boldsymbol{\alpha}_t \right)^q \right)^{\frac{2}{q}} + \frac{1}{8\gamma_0^2 \eta} \left(\sum_{m=1}^{M} \left(\boldsymbol{\alpha}^* \mathbf{Q}^{(m)} \boldsymbol{\alpha} \right)^q \right)^{\frac{2}{q}} \\ &-\sum_{t=1}^{T} \mathbf{s}_t^* \boldsymbol{\alpha}_t \qquad\qquad s.t.\ \forall \hat{x}_t^{ij} \in S : 0 \leq \alpha_{ij}^t \leq \frac{C}{N}\end{aligned} \tag{4}$$

where $1/p+1/q=1$ and $s_t^{ij} = \left(\sigma_t^{ij} - \delta_t^{ij} d_t^{ij} \right)$. $\mathbf{Q}^{(m)}$ and $\mathbf{Q}_{t,t}^{(m)}$ is represented by:

$$\begin{aligned}Q_{t,t'}^{(m)}(ij,kl) = tr\left(\delta_t^{ij} \delta_{t'}^{kl} \left(\hat{x}_t^{ij,m} \right)^* \hat{x}_{t'}^{kl,m} \right) \\ Q_{t,t}^{(m)}(ij,kl) = tr\left(\delta_t^{ij} \delta_t^{kl} \left(\hat{x}_t^{ij,m} \right)^* \hat{x}_t^{kl,m} \right)\end{aligned} \tag{5}$$

From (2) and (4) we see that the tradeoff parameters γ_t, t=0,…,T control the regularization of $A_t^{(m)}$ and the information sharing structure. When γ_t → ∞ and γ_t is small for t=1,…,T, $A_0^{(m)}$ become zeros and the tasks become more independent. When γ_0 is small and $\gamma_t \to \infty, t = 1,...,T$, the $A_t^{(m)}$ for all the tasks will be **0** and they will be merged into one single task. For other case, when $\gamma_t = 0, t = 1,...,T$, our model reduces to a single task, and when $\gamma_0 = 0$, our model to reduce to T independent tasks. When $\gamma_t, t = 0,...,T$ is not zero, the larger γ_0 or γ_t is, the less influence of the corresponding task(s) will be brought to the final model. Therefore, we can take advantage of this property to control the information sharing among tasks.

From (3) we see that our model is in fact a sample sharing multi-task learning. Samples from other tasks will be served as extra support vectors for each single task, where their influence can be controlled by $\gamma_t, t = 0,...,T$. Therefore, for visual applications on small dataset, incorporating learning tasks using Web data with social tagging will endow the application with more available data resources, thus the performance is likely to be enhanced.

The dual objective problem (4) is convex with respect to $\boldsymbol{\alpha}$, if and only if all the matrices $\mathbf{Q}^{(m)}$ and $\mathbf{Q}_{t,t}^{(m)}$ are positive semi-definite. This requirement can be easily satisfied for most similarity such as inner product and RBF kernels.

3.2. Kernelization

The learned similarity can be easily kernelized by replacing all $\mathbf{x}$ by $\phi(\mathbf{x})$, where ϕ is the feature map corresponding to any given kernel. We denote the original kernel for each feature channel as $K^{(m)}$, then the learned kernel for each task is given by:

$$\begin{aligned}&\widetilde{K}_t(x_a, x_b) = -\sum_{m=1}^{M} \sum_{t'=1}^{T} \sum_{ij \in T_{t'}} \alpha_{t'}^{ij} \delta_{t'}^{ij} \left(\frac{b_0^{(m)}}{\gamma_0} + \frac{\mathbf{1}_t(t') \bullet b_t^{(m)}}{\gamma_t} \right) \bullet \\ &\left(K^{(m)}(x_a, x_t^i) - K^{(m)}(x_a, x_t^j) \right)\left(K^{(m)}(x_b, x_t^i) - K^{(m)}(x_b, x_t^j) \right)\end{aligned} \tag{6}$$

Where $\mathbf{1}_{\text{t}}(t')$=1 if t'=t, else it is zero.

3.3. Training data selection

It has been discussed in [25], that the neighborhood of each data is most influential to the model. This heuristic reduces the training pairs from O(N^2) scale to O(N) scale. To identify the neighborhood for each training data quickly, we adopt Locality Sensitive Hashing [4] for linear metric learning. For nonlinear metric, the Kernel LSH [22] is built on the average kernel representation.

We generate L_H hash functions so that each data item is mapped into a binary code vector with length L_H. Then they are hashed into a set of hash buckets. We construct three hash tables to improve the recall of the true neighborhoods. For neighborhood search of each query, we select top θ_S nearest samples with same class label as the query, and top θ_D nearest samples with different class labels. We empirically set $L_H = 40$, $\theta_S = 3$, $\theta_D = 6$ for each task. The number of unique hash buckets is usually 1/3~1/2 of the original data size and the maximum number of items within each bucket will not exceed 20, which means that the data is distributed dispersedly. Therefore, the computational cost for neighborhood search can be reduced significantly.

3.4. Category grouping strategy

In our study, each learning task may correspond to a learned metric which discriminates a group of categories from others. For example, when we get the group structure (C_1, C_2, (C_3, C_4)), we learn 3 metrics where one for classifying C_1, one for C_2 and one for both C_3 and C_4. The grouping of categories be done with different schemes, one is natural grouping according to the existing hierarchical semantic structure such as *WordNet*, or by hierarchical grouping according to the overall similarity on multiple feature representations among different classes. Another potential way is similar with automatic grouping by minimizing some objective function [20], which may requires huge computational effort for hundreds of classes since it has only be tested on small dataset with dozens of classes. With the hierarchical category grouping structure, one can determine how many metrics should be learned with our method, thus a better tradeoff between efficiency and accuracy can be achieved.

In this paper, we adopt the hierarchical visual grouping as it better explores visual correlations and achieves better data balance among different learning tasks. We calculate the average multiple feature representations on the images from each class, so that each semantic class will have an average image feature on the concatenated feature representation. Then hierarchical clustering is conducted on these feature representations. We use this hierarchical structure to form the tasks, so that each level corresponds to different number of tasks. We will provide comparison on several grouping techniques in Section 4.3.

3.5. Optimization

Since the dual problem (4) is differentiable with respect to $\boldsymbol{\alpha}$, we propose an efficient solver based on the coordinate gradient descent method (CGD) [16] that was successfully used in optimizing convex problems such as SVM. We minimize the dual problem in (4) by decomposing it into a series of one-variable differentiable convex sub-problems with respect to each α_t^{ij}. The advantage of CGD is small computational cost for each step and fast convergence rate.

Algorithm 1: Coordinate Gradient Descent Solver

```
Input: training pair set $\mathbf{X}^{IJ}$
$k=1$, $U_b \leftarrow +\infty$, $L_b \leftarrow -\infty$, $\overline{\mathbf{X}}^{IJ} \leftarrow \mathbf{X}^{IJ}$
While $k{+}{+} < k_{max}$ or Not Convergent do:
  (1) $\overline{\mathbf{X}}^{IJ} \leftarrow RandPerm(\overline{\mathbf{X}}^{IJ})$, $\overline{U}_b \leftarrow -\infty, \overline{L}_b \leftarrow +\infty$
  (2) For each $\alpha_t^{ij}$, $ij_t \in \overline{\mathbf{X}}^{IJ}$
      (a) Calculate Gradient $\nabla R(\boldsymbol{\alpha})_t^{ij}$
      (b) If ( $\alpha_t^{ij} = 0$ & $\nabla R(\boldsymbol{\alpha})_t^{ij} > U_b$ ) || ( $\alpha_t^{ij} = C_t^{ij}$ & $\nabla R(\boldsymbol{\alpha})_t^{ij} < L_b$ )
              $\overline{\mathbf{X}}^{IJ} \leftarrow \overline{\mathbf{X}}^{IJ} / ij_t$, continue.
          Else get $\nabla^P R(\boldsymbol{\alpha})_t^{ij}$ using (9).
      (c) $\overline{U}_b \leftarrow \max(\overline{U}_b, \nabla^P R(\boldsymbol{\alpha})_t^{ij}), \overline{L}_b \leftarrow \min(\overline{L}_b, \nabla^P R(\boldsymbol{\alpha})_t^{ij})$
      (d) Calculate $H_t^{ij,ij}$, Update $\alpha_t^{ij}$ by (10), $o_t^m$ by (11)
  (3) If $\overline{U}_b - \overline{L}_b < \varepsilon$
    If $\overline{\mathbf{X}}^{IJ} = \mathbf{X}^{IJ}$, BREAK. Else $\overline{\mathbf{X}}^{IJ} \leftarrow \mathbf{X}^{IJ}, U_b \leftarrow +\infty, L_b \leftarrow -\infty$
  (4) If $\overline{U}_b < 0$ then $U_b \leftarrow +\infty$. Else $U_b \leftarrow \overline{U}_b$
  (5) If $\overline{L}_b > 0$ then $L_b \leftarrow -\infty$. Else $L_b \leftarrow \overline{L}_b$
Output: $\boldsymbol{\alpha}$, $\boldsymbol{b}$
```

Suppose the training pairs of t^{th} task is denoted by $\mathbf{X}_t^{IJ}$, and the whole set denoted by $\mathbf{X}^{IJ} = \mathbf{X}_1^{IJ} \cup \ldots \cup \mathbf{X}_T^{IJ}$. The process starts from an initial point $\boldsymbol{\alpha}^0 \in R^{|\mathbf{X}_{IJ}|}$ and generates a sequence of vector $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$. We define the process from $\boldsymbol{\alpha}^k$ to $\boldsymbol{\alpha}^{k+1}$ as the outer-iteration. Each outer-iteration has $|\mathbf{X}^{IJ}|$ inner-iterations where each corresponds to a sub-problem with respect to α_t^{ij} as:

$$\min_d R(\boldsymbol{\alpha} + d\mathbf{e}_t^{ij}), \; s.t. \; 0 \le \alpha_t^{ij} \le C_t^{ij} \tag{7}$$

where $\mathbf{e}_t^{ij} = [0,..,1,...0]^*$. The objective function in (7) is a quadratic function of d:

$$R(\alpha_t^{ij} + d) = const + \frac{1}{2} H_t^{ij,ij} d^2 + \nabla R(\boldsymbol{\alpha})_t^{ij} \tag{8}$$

where $\nabla R(\boldsymbol{\alpha})_t^{ij}$ and $H_t^{ij,ij}$ denote the gradient and Hessian respectively. Since each α_t^{ij} is bounded to $[0, C_t^{ij}]$, we need to find a projected gradient calculated by:

$$\nabla^P R(\boldsymbol{\alpha})_t^{ij} = \begin{cases} \min\left(\nabla R(\boldsymbol{\alpha})_t^{ij}, 0\right), \alpha_t^{ij} = 0 \\ \max(\nabla R(\boldsymbol{\alpha})_t^{ij}, 0), \alpha_t^{ij} = C_t^{ij} \\ \nabla R(\boldsymbol{\alpha})_t^{ij}, 0 < \alpha_t^{ij} < C_t^{ij} \end{cases} \tag{9}$$

If the projected gradient is not zero, we update α_t^{ij} by:

$$\alpha_t^{ij} \leftarrow \min\left(\max\left(\alpha_t^{ij} - \nabla R(\boldsymbol{\alpha})_t^{ij} / H_t^{ij,ij}, 0\right), C_t^{ij}\right) \tag{10}$$

Then we update o_t^m as:

$$o_t^m \leftarrow o_t^m + 2d \bullet tr\left(B_t^{(m)} \delta_t^{ij} \hat{x}_t^{ij,m}\right) + d^2 Q_{t,t}^{(m)}(ij, ij) \tag{11}$$

Besides, we adopt two acceleration schemes proposed by Hsieh *et al.* [16], random permutation of sub-problems and coefficient shrinking. Details are shown in Algorithm 1.

4. Experiments

Our experiments are conducted on a database which is composed on 3 benchmark datasets, where two of them are VOC 2007 [10] and ImageNet-250 subset from ImageNet database [6] with clean semantic labels, and MFLICKR [17] with 1 million images and thousands of different social tags. The ImageNet-250 is an image subset selected from the large scale ImageNet data corpus covering many common visual concepts including categories from *vehicle*, *plant*, *animal*, *food*, *instrument*, *scene* and *sports*. Each image category in ImageNet-250 contains more than 500 images. For MFLICKR, we select the top 300 most frequently used tags in this work. We calculate 9 types of features and kernels for all the images, namely, spatial pyramid with linear and chi-square kernels on HOG, LBP, Self Similarity, dense gray SIFT and dense color SIFT, as well as the other with linear and RBF kernels on Geometric Blur, GIST, block Color Moment and wavelet texture histograms.

For VOC'07 data, we use train-val and test data and the Mean Average Precision (MAP). For ImageNet-250, each category is randomly split 60%~40% into a set of the training and testing data, which results in about 140K training images and 99K testing images. For evaluation on this dataset, we adopt the Mean Accuracy (MA) as in [6] which records the average percent of correct predictions among all the classes. For MFLICKR data, we select 100K for training, and 50K images for testing. We record the average per-tag precision and recall at top 100 results. For performance evaluation on all the methods, including ours and other approaches, we use *k*-NN as the learning and predicting strategy. For evaluation of MAP and per-tag precision and recall on MFLICKR, we rank the results by the learned similarity measures with respect to the query and retrieved samples. In the rest of Section 4, we denote linear and kernelized multi-task multi-feature similarity learning method as M^{2}SL-L and M^{2}SL-K, respectively.

4.1. Efficiency

We conduct some analysis on the optimization issue to show the efficiency of our method. We implemented our work in Matlab on a desktop with Intel i5 2.8GHZ dual core CPU and 8G RAM. We randomly select 30 classes from ImageNet-250. We evaluate 5 methods, where *LCGD* and *LACC* denote the optimization without and with acceleration for the multi-task linear similarity learning, and *KCGD* and *KACC* denote the optimization without and with acceleration for kernelized multi-task similarity learning respectively. We compare our method with multi-task *LMNN* [28] with concatenated feature. For all the method, we decompose the classes into 4 tasks on the average concatenated feature representation of each class. We set p=2.5, $C / N = 8$, $\gamma_0 = 2$, $\gamma_t = 1$, σ_t^{ij} =0.4M. We record training time (T), number of outer iterations (#OI), average number of reduced coefficients (#AR) and ratio of support vectors (SV) and Mean Accuracy (MA) in Table 1.

From the results we see that *LCGD* achieves good performance in training time. With acceleration techniques, *LACC* will be much faster without performance degradation. The kernelized similarity learning (*KCGD* and *KACC*) spend about 50% more time on training, because the kernel calculation is more time consuming compared with inner product. However, the model with less support vectors is obtained, which is more preferred in many real world applications. In fact, the training time of *KACC* can still be optimized with acceleration scheme such as kernel caching. The training time of multi-task *LMNN* is a lot more than any of our method, as it does not outperform our approach. This observation reveals that metric learning on high dimensional data may take great risk of over-fitting.

Table 1: Statistics of the training procedures

	T(sec)	#**OI**	#**AR**	**SV**(%)	**MA**
LCGD	11530	258	--	0.329	0.312
LACC	7962	233	614	0.293	0.313
KCGD	17685	216	--	0.154	0.366
KACC	12296	159	879	0.143	0.365
LMNN	43876	--	--	--	0.301

4.2. Different l_p norm

We analyze the influence of p in our model, where different settings of p will influence the physical structure of the dual problem in (4). We evaluate the performance on VOC'07 validation data on different setting as p = (1.05, 1.33, 2, 2.5, 3, 4), and the other parameter setting is identical to Section 4.1, the relative performance (RP) which is the performance divided by the highest from all the parameter settings is demonstrated in Table 2. We notice that p=2.5 performs consistently best. The performance decreases when p is large, say p=4, which leads to more dense kernel weight values and the performance will tend to be similar with the average kernel.

Table 2: The relative performance change with different p

p	1.05	1.33	2.0	2.5	3	4
RP	0.961	0.973	0.98	1.00	0.969	0.954

To better show how different p influence the learned kernel weights, we further shows the learned weights using the kernelized method for two different classes in VOC'07, namely, the *car* class and *pottedplant* class in Figure 1. When p is close to 1, only one or two kernels will dominate the contribution to the model. For *car* class, the weight of kernel calculated on HOG is larger than other features, meaning that HOG feature is good at discriminating *car* from other classes. For the *pottedplant* class, it seems that color descriptor is more discriminative.

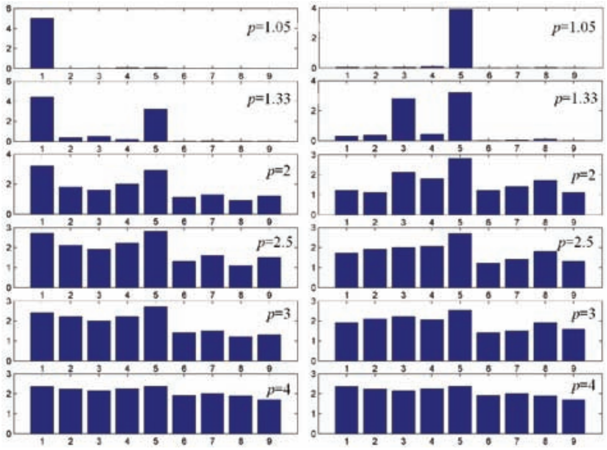

Figure 1. The learned feature weight of *car* class (left) and *pottedplant* class (right). The bars from left to right in each sub figure corresponds to HOG, LBP, SS, gray SIFT, color SIFT, GB, GIST, CM and Wavelet

4.3. Comparison of grouping strategies

We compare three different category grouping strategy, namely, random grouping (**RG**), grouping with *WordNet* distance (**WD**) [7] and grouping with hierarchical visual clustering (**VC**). The model parameter setting is the same as in Section 4.1. We report the Mean Accuracy of M^{2}SL-K with respect to different number of tasks on ImageNet-250 data in Table 3. We can see that visual clustering obtains better results compared with other two. One explanation for the result is the "discriminative part" of the learned metric for one visually similar category group tends to be better generalized to identify those common feature subsets that can distinguish this category group from others. Another observation from Table 3 is that when the number of tasks is small, using different grouping strategies will lead to larger performance gap.

Table 3: Effect of different grouping strategies

#tasks:	4	8	16	31
RG	0.298	0.352	0.394	0.423
WD	0.323	0.376	0.412	0.429
VC	0.352	0.397	0.429	0.445

4.4. Information sharing among tasks

We study how our multi-task metric learning improves the performance with different number of tasks and information sharing among them. To this end, we conduct experiment on ImageNet-250 datasets, and test the linear and kernel version of our method. For this part, we set p=2.5, C / N = 8 and σ_t^{ij} =0.4M for both linear and kernelized learning of our models. For M^{2}SL-L and M^{2}SL-K, we set $\gamma_0 = 2$ and $\gamma_t = 1$.

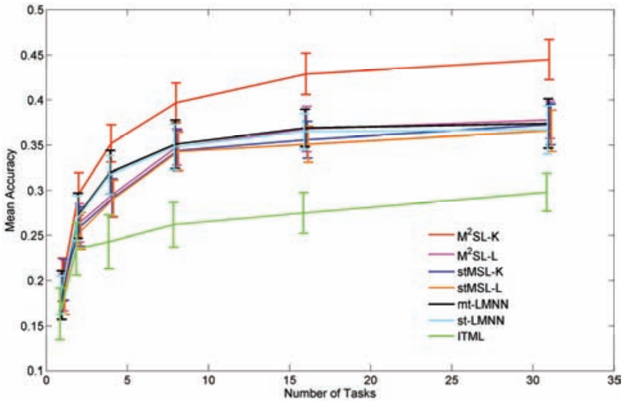

Figure 2: The performance curve with different number of tasks on ImageNet-250

We refer to our metric learning where γ_0=0 and γ_t = 1 as single task metric learning (stMSL-L and stMSL-K), as there is no information sharing among tasks. We compare our methods with the single task LMNN [33] (st-LMNN) where each task learns a model without information sharing, and multi-task LMNN [28] (mt-LMNN) where we also set γ_0 = 2 and γ_t = 1 for fair comparison. Besides, we also test ITML [5]. The performance curves with different number of tasks are shown in Figure 2.

From Figure 2, we observe that when the number of tasks increases, the performance for all the methods increases. Specifically, methods with information sharing (mt-LMNN and M^{2}SL-K) achieve better performance and smaller variances. Since we can expect the results tend to be better when the number of tasks exceeds 31, we do not evaluate the results because of significant increase on model training. Despite this, the performance for learning with 31 tasks can still demonstrate the advantage of our method.

4.5. Visual categorization

We conduct experiments to compare our method with several state-of-the-art approaches on visual semantic categorization. The model parameter setting is the same as in Section 4.1. We compare our method with 7 approaches on both VOC'07 and ImageNet-250 dataset: (1) EUC: the Euclidean metric on the concatenated feature. (2) EUC-PCA: the Euclidean metric on the concatenated feature representation after dimensional reduction using PCA (Dim: 500). (3) ITML [5]: using features in (2). (4) LFDA [29]: localized FDA on features in (2). (5) st-LMNN: single task LMNN [33] which learns the *Mahalanobis* matrices for each class on features in (2). (6) mt-LMNN: Multi-task LMNN [28] which learns the *Mahalanobis* matrices in multi-task learning framework on features in (2). (7) NCA [15]: run on features as in (2).

For VOC'07, we set the number of tasks as 20 for both M^{2}SL-L and M^{2}SL-K, where each class corresponds to a learned metric. For ImageNet, we set the task number as 31.

The results are recorded in Table 4. From Table 4 we see that our M^{2}SL-K achieves the highest performance on both datasets. We see from both the performance of st-LMNN and our approaches that multiple metrics are indeed helpful for enhancing the performance. However, without information sharing among tasks, the single task metric learning does not outperform mt-LMNN and our approaches M^{2}SL-L and M^{2}SL-K on VOC'07 data. Our linear version M^{2}SL-L slightly underperforms mt-LMNN and st-LMNN on ImageNet-250 dataset. The reason may be that we only discover the feature correlation inside each feature, while st-LMNN and mt-LMNN fully discovers the inter-correlation among different features. However, M^{2}SL-K achieves the best performance because it is capable of using the nonlinear kernels. In general, from the experimental comparison we can see that our methods achieve promising results on semantic categorization.

Table 4: The MAP on VOC'07 and MA for ImageNet-250

Methods	VOC'07	ImageNet-250
EUC	0.181	0.192
EUC-PCA	0.296	0.264
ITML	0.398	0.298
LFDA	0.384	0.305
NCA	0.415	0.315
st-LMNN	0.569	0.367
mt-LMNN	0.572	0.374
M^{2}SL-L	0.577	0.378
M^{2}SL-K	**0.603**	**0.445**

4.6. Categorization with social tagging

We study whether incorporating the social tagging data can boost the performance of semantic categorization. We conduct the semantic categorization as main tasks on both VOC'07 and ImageNet-250. For the auxiliary learning tasks on social image data, we automatically group 4 tags in one task. For VOC'07 we select 50 tags which are semantically related with the 20 semantic classes as the candidate auxiliary tasks. For ImageNet-250, all the 300 tags are used to form the candidate auxiliary tasks. We set γ_0=2, γ_t = 1 of the main tasks, and γ_t = 5 of the auxiliary tasks in order to suppress the influences of noise contained in social tagging information. We test both M^{2}SL-L and M^{2}SL-K for both datasets. The number of main tasks for both methods is 20 for VOC'07 and 31 for ImageNet-250 respectively. We conduct experiments on these two datasets for semantic categorization by varying the numbers of the auxiliary tasks selected from the candidate auxiliary tasks set. The experimental results are shown in Figure 3.

We can see that when the number of auxiliary tasks increases, the performance of our methods on both VOC'07 and ImageNet-250 are improved. However, when the information transferred from the auxiliary tasks overwhelm the main tasks, the performance will be likely to degrade due to the spread of noisy information existing in social tagging. For VOC'07, when the number of auxiliary tasks become large, more social tagging images with less related semantic correlation as well as noisy information will be introduced. Under this situation, instead of getting help from the auxiliary tasks, the model is more likely to corrupt.

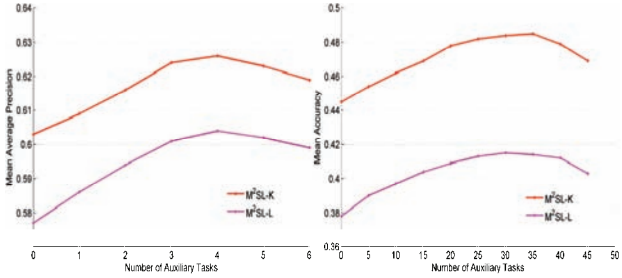

Figure 3: The performance with different numbers of auxiliary tasks on VOC'07(left) and ImageNet-250(right)

4.7. Tagging with semantic information

We study the task of image tagging with the help of semantic information. We conduct experiments on the 50K test data selected from MFLICKR dataset. We test two versions of our methodand two baseline approaches EUC-PCA and mt-LMNN on the reduced concatenated feature representation. We set γ_0=1, γ_t = 1 for the main tasks, and γ_t = 2 for the auxiliary tasks. The other parameter setting is the same as Section 4.1. From MFLICKR dataset, we also automatically group 4 tags for each task for both our method and mt-LMNN. We select the ImageNet-250 as the dataset with semantic labels, and the number of task is also 31 as in previous sections. The performance is recorded in Table 5. Our methods significantly outperform the baseline methods in terms of Precision. Moreover, our M^{2}SL-K achieves the highest performance on both Precision and Recall. The Recall of mt-LMNN slightly outperforms our linear similarity learning method.

Table 5: The performance of automatic image tagging

	EUC-PCA	mt-LMNN	M^{2}SL-L	M^{2}SL-K
Prec	0.26	0.48	0.58	**0.61**
Rec	0.28	0.52	0.50	**0.59**

4.8. Positive semi-definiteness

Our method does not guarantee the positive semi-definiteness of the overall learned metric, *i.e.*, the positive semi-definiteness of the metric for each feature channel. However, in many applications the results are usually ranked by their relative value, the positive semi-definiteness is not a necessary requirement. We can also project the metric into the positive semi-definite space [33]. As for the experiments in this paper, we have not obtained any metric with negative Eigen-values after training, although we do observe some of them during the model training. In future study, we will conduct more experiments on the influence of the projection.

5. Conclusion

We propose a novel multi-task multi-feature similarity learning method which learns a set of overall similarity measure for multiple tasks simultaneously. Our method demonstrates more flexibility to explore the intrinsic model sharing and feature weighting relations on image data with large amount of classes without significant increase of the number of models. Our model allows knowledge transfer among semantic labels and social tagging information which combines the information fusion from both sides for effective image understanding. We will study how to make better use of the hierarchical semantic structure for large scale visual applications in future works.

6. Acknowledgements

This work was supported in part by National Basic Research Program of China (973 Program): 2012CB316400, in part by National Natural Science Foundation of China: 61025011, 60833006 and 61133003, and in part by Beijing Natural Science Foundation: 4111003. It was also was supported in part to Dr. Qi Tian by ARO grant W911BF12-1-0057, NSF IIS 1052851, Faculty Research Awards by Google, FXPAL, and NEC Laboratories of America, respectively.

7. References

[1] A. Argyrious, T. Evgenious and M. Pontil. Multi-task Feature Learning. *NIPS*, 2006.

[2] F. Bach, G. Lanckriet, and M. Jordan. Multiple Kernel Learning, Conic Duality, and the SMO Algorithm. *ICML*, 2004.

[3] R. Caruana. Multi-task Learning. *Machine Learning*, 28: 41-75, 1997.

[4] M. Charikar. Similarity Estimation Techniques from Rounding Algorithms. *ACM Symposium on Theory of Computing*, 2002.

[5] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic Metric Learning. *ICML*, 2007.

[6] J. Deng, A. Berg, K. Li and L. Fei-Fei. What Does Classifying More Than 10,000 Image Categories Tell Us? *ECCV*, 2010.

[7] J. Deng, A. Berg and L. Fei-Fei. Hierarchical Semantic Indexing for Large Scale Image Retrieval. *CVPR*, 2011.

[8] L. Duan, D. Xu, I. W. Tsang, and J. Luo. Visual Event Recognition in Videos by Learning from Web Data. *CVPR*, 2010.

[9] L. Duan, I. W. Tsang and D. Xu. Domain Transfer Multiple Kernel Learning. To appear in *PAMI.*

[10] M. Everingham, A. Zisserman, C. K. I. Williams and L. Van Gool. The Pascal Visual Object Classes Challenge 2007 Results. *Technical report*, 2007.

[11] T. Evgeniou and M. Pontil. Regularized Multi-task Learning. *ACM KDD*, 2004.

[12] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning Globally-Consistent Local Distance Functions for Shape Based Image Retrieval and Classification. *ICCV*, 2007.

[13] P. Gehler and S. Nowozin. On Feature Combination for Multiclass Object Classification. *ICCV*, 2009.

[14] A. Globerson, S. Roweis. Metric Learning by Collapsing Classes. *NIPS*, 2006.

[15] J. Goldberger, S. Roweis, G. Hinton, R. Salakhutdinov. Neighborhood Component Analysis. *NIPS*, 2005.

[16] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi and S. Sundararajan. A Dual Coordinate Descent Method for Large Scale Linear SVM. *ICML*, 2008.

[17] M. J. Huiskes and M. S. Lew. The MIR Flickr Retrieval Evaluation. *ACM MIR* 2008, Vancouver, Canada.

[18] S. Hwang, K. Grauman and F. Sha. Learning a Tree of Metrics with Disjoint Visual Feature. *NIPS*, 2011.

[19] S. Hwang, F. Sha and K. Grauman. Sharing Features between Objects and Their Attributes. *CVPR*, 2011.

[20] Z. Kang, K. Grauman and F. Sha. Learning with Whom to Share in Multi-task Feature Learning. *ICML*, 2011.

[21] S. Kim, A. Magnani, S. Boyd. Optimal Kernel Selection in Kernel Fisher Discriminant Analysis. *ICML*, 2006.

[22] B. Kulis, and K. Grauman. Kernelized Locality Sensitive Hashing for Scalable Image Search. *ICCV*, 2009.

[23] B. Kulis, M. Sustik, and I. Dhillon. Low-Rank Kernel Learning with Bregman Matrix Divergences. *JMLR*, 10: 341-376, 2009.

[24] A. Kundu, V. Tankasali, C. Bhattacharyya, and A. Ben-Tal. Efficient Algorithms for Learning Kernels from Multiple Similarity Matrices with General Convex Loss Functions. *NIPS*, 2010.

[25] J. T. Kwok, I. W. Tsang. Learning with Idealized Kernels. *ICML*, 2003.

[26] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui and M. Jordan. Learning the Kernel Matrix with Semi-definite Programming. *JMLR*, 5: 27–72, 2004.

[27] Y. Lin, T. Liu and C. Fuh. Multiple Kernel Learning for Dimensionality Reduction. *PAMI,* 33(6): 1147-1160, 2010.

[28] S. Parameswaran, K. Q. Weinberger. Large Margin Multi-Task Metric Learning. *NIPS*, 2010.

[29] M. Sugiyama. Dimensionality Reduction of Multimodal Labeled Data by Local Fisher Discriminant Analysis. *JMLR*, 8, 1027-1061, 2007.

[30] A. Torralba. K. Murphy and W. Freeman. Sharing Visual Features for Multi-Class and Multi-View Object Detection. *PAMI*, 29(5): 854-869, 2007.

[31] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple Kernels for Object Detection. *ICCV*, 2009.

[32] S. V. N. Vishwanathan, Z. Sun, N. Theera-Ampornpunt and M. Varma. Multiple Kernel Learning and the SMO Algorithm. *NIPS*, 2010.

[33] K. Q. Weinberger, L. K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *JMLR*, 10: 207-244, 2009.

[34] X. Yuan and S. Yan. Visual Classification with Multi-Task Joint Sparse Representation. *CVPR*, 2010.