Sparse Online Relative Similarity Learning

Dezhong Yao*, Peilin Zhao†, Chen Yu*, Hai Jin* and Bin Li‡
*Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
†Data Analytics Department, Institute for Infocomm Research, A*STAR, 138632, Singapore
‡Department of Finance, Economics and Management School, Wuhan University, Wuhan 430072, China
Email: {dyao, yuchen, hjin}@hust.edu.cn, zhaop@i2r.a-star.edu.sg, binli.whu@whu.edu.cn

***Abstract*—For many data mining and machine learning tasks, the quality of a similarity measure is the key for their performance. To automatically find a good similarity measure from datasets, metric learning and similarity learning are proposed and studied extensively. Metric learning will learn a Mahalanobis distance based on positive semi-definite (PSD) matrix, to measure the distances between objectives, while similarity learning aims to directly learn a similarity function without PSD constraint so that it is more attractive. Most of the existing similarity learning algorithms are online similarity learning method, since online learning is more scalable than offline learning. However, most existing online similarity learning algorithms learn a full matrix with d^2 parameters, where d is the dimension of the instances. This is clearly inefficient for high dimensional tasks due to its high memory and computational complexity. To solve this issue, we introduce several Sparse Online Relative Similarity (SORS) learning algorithms, which learn a sparse model during the learning process, so that the memory and computational cost can be significantly reduced. We theoretically analyze the proposed algorithms, and evaluate them on some real-world high dimensional datasets. Encouraging empirical results demonstrate the advantages of our approach in terms of efficiency and efficacy.**

I. Introduction

In many applications, it is important to have a suitable similarity measure between objectives, such as text mining [1], image retrieval [2] and so on. For example, when using k nearest neighbor (k-NN) to classify documents, the accuracy will significantly depend on the quality of the similarity measure. In addition, the performance of content based image retrieval also heavily relies on the similarity measure. To find a meaningful similarity measure, online metric learning has been extensively studied for years [3], [4], [5], [6] due to the scalability of online learning [7], [8], [9], [10], where a Mahalanobis distance function is learnt. Specifically, online metric learning approaches try to learn a linear Mahalanobis distance [11], [12], [13], [14], satisfying a set of constraints. Generally the constraints assign larger similarity scores for pairs of similar instances than dissimilar instances. This is equivalent to learning a linear projection of the data to a feature space where constraints on the training set are better satisfied. These metric learning algorithms have significantly improved the classification accuracy of k-NN and retrieval performance of content-based image retrieval on real-world datasets. As we know, Mahalanobis distance requires the model, i.e. the matrix, to be positive semi-definite (PSD). So during the learning process, projections are needed to keep the model in the PSD cone. However, the projections need singular value decompositions with very high time complexity, which makes metric learning not scalable with respect to the dimension of instances.

To solve this issue, online similarity learning has been recently studied [15], [13], [16], [17], [14], [18], which does not require its model to be PSD, so it is much more efficient than metric learning. In addition, the similarity metric can even be non-symmetric, so it can be applied to measure similarities of objects from two different feature spaces. Following this principle, similarity learning is studied for classification problems [15], [19], and image retrieval tasks [13], [14]. Empirical studies show that those online similarity learning algorithms generally are better than or at least comparable with the metric learning algorithms on the classification tasks and image retrieval problems. At the same time, they are more efficient and scalable than metric learning algorithms, since the projection steps are avoided. As a result, online similarity learning is more suitable to large-scale data mining and machine learning tasks. However, these existing online similarity learning algorithms [13], [20] usually try to learn a full matrix during the learning process, which results in high memory and computational cost for high dimensional learning tasks.

Recently, to solve high dimensional metric/similarity learning problems, there are several off-line learning algorithms that have focused on learning sparse global linear metrics, e.g., SDML [21], SML [22] and GSML [23]. They are all based on Davis's work [24] and try to learn a global sparse model by minimizing the Bregman divergence between the model and the identity matrix corresponding to the Euclidean distance, subject to a set of linear constraints. By minimizing Bregman divergence, the learned Mahlanobis matrix usually can be as close to the identity matrix as possible [21], so that the learnt Mahlanobis matrix can be very sparse, since the identity matrix is rather sparse. However, these algorithms cannot handle online learning scenario, which is more realistic for many real-world data stream tasks. Motivated by the above observations, we propose a new *Sparse Online Relative Similarity Learning* (**SORS**) scheme, where the similarity function is a sparse model learnt efficiently and scalably from the training data. Specifically, we proposed two types of algorithms: SORS based on recent proximal online gradient descent; and AdaSORS based on state-of-the-art adaptive proximal online gradient descent. The scalability and efficiency of these advanced sparse online learning strategies make the proposed algorithms attractive to large-scale high dimensional real world applications. The contributions of our work are summarized as following: 1) We introduce a new Sparse Online Relative Similarity learning scheme that takes sparsity into consideration; 2) Based on

the scheme, we proposed several sparse online similarity learning algorithms; 3) We theoretically analyze the proposed algorithm; 4) We apply the proposed algorithms on a set of real-world datasets, where encouraging empirical results are achieved compared with some state-of-the-art similarity/metric learning algorithms.

The rest of this paper is organized as follows. The related work is reviewed in the next section. Then, section 3 presents our proposed sparse online relative similarity learning approaches. Section 4 conducts an extensive set of experiments by comparison with several state-of-the-art methods, and section 5 concludes our paper.

II. Related Work

A. Similarity/Metric Learning

Online similarity learning is a group of efficient and scalable machine learning algorithms [25], [10], [14]. Most existing works in similarity learning relies on learning a Mahalanobis distance, which has been found to be a sufficiently powerful class of metrics that work on many real-world problems. POLA [26], Pseudo-metric Online Learning Algorithm, is the first online Mahalanobis distance learning approach which learns the distance metric $M \succeq 0$ as well as a threshold $b \geq 1$. The authors provide a regret bound for this algorithm, under the assumption that the dataset can be separated by some metric. The work LMNN by [5] addresses the distance learning problem by exploring a large margin nearest neighbor classifier LEGO [12], LogDet Exact Gradient online, is an improved version of POLA based on LogDet divergence regularization. It features tighter regret bounds, more efficient updates, and automatic semi-definiteness. RDML [27], Regularized Distance Metric Learning, is also similar to POLA but is more flexible. At each step t, instead of forcing the margin constraint to be satisfied, it performs a gradient descent step. Those methods focus on the symmetric distance: measure the distance between two images $\mathbf{x}_1$ and $\mathbf{x}_2$ by $(\mathbf{x}_1 - \mathbf{x}_2)^\top M(\mathbf{x}_1 - \mathbf{x}_2)$, where the matrix M must be positive semi-definite.

The proposed online learning scheme is close to the recent work of scalable image similarity learning (OASIS) [13], [14]. This method learns bilinear similarity with a focus on large-scale problems. In this model, given two images $\mathbf{x}_1$ and $\mathbf{x}_2$, it measures similarity through $\mathbf{x}_1^\top M\mathbf{x}_2$. The metric M is related to the (generalized) cosine similarity but does not include normalization nor PSD (symmetric positive semi-definite) constraint. In another part, unlike the Mahalanobis distance, it can define a similarity measure between instances of different dimensions. However, the learnt model M is not sparse, which is unsuitable to large scale tasks for high dimension datasets.

B. Sparse Online Learning

The general online learning algorithms have solid performance on classification problems. However, for large-scale high-dimensional task, they learn a full feature classifier, which may cause high memory demand and heavy computational load. To deal with this limitation, the *Sparse Online Learning* [28], [29], [30] has been studied recently. The goal of sparse online learning is to learn a sparse classifier, which only contains limited active features. In current *Sparse Online Learning* studies, there are two types of solutions. One solution follows the general idea of subgradient descent with truncation. Such as, FOBOS method by [29], makes an improvement based on the *Forward-Backward Splitting* method to solve the sparse online learning problem. There are two steps in the FOBOS method: (1) an unconstrained subgradient descent step, and (2) an instantaneous optimization step for a trade off between maintaining the result calculated in the first step and minimizing ℓ_1 norm regularization. The optimization goal in the step 2 can be solved by adopting *soft-thresholding* operations, which make truncation on the weight vectors. Under the guidance of this idea, the TG, Truncated Gradient, scheme [28] is proposed by Langford et al.. When there are less than a predefined threshold θ, the TG scheme truncates coefficients every K steps. This demonstrates that truncation on each iteration is too aggressive as each step modifies the coefficients by only a small amount. The second solution of sparse online learning uses the dual averaging method [31], which studies the regularization structure in an online setting. RDA [32], Regularized Dual Averaging, is one representative study, which simultaneously achieves the optimal convergence rates for both convex and strongly convex loss. An extension work [33] of RDA algorithm is presented by using a more aggressive truncation threshold and generates significantly more sparse solutions. However, all the above algorithms are designed for classification problems.

III. Sparse Online Relative Similarity Learning

In this section, we will review the problem setting, propose our algorithm, and provide theoretical guarantees.

A. Problem Formulation

Following [13], we would study the problem of online similarity learning. Our goal is to learn a similarity function $S : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ based on a sequence of training triplets $\{(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-) \in \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d | t \in [T]\}$ with $[T] = \{1, \ldots, T\}$, where the relevance between $\mathbf{x}_t$ and $\mathbf{x}_t^+$ is greater than that between $\mathbf{x}_t$ and $\mathbf{x}_t^-$. Specifically, we would like to learn a similarity function $S(\mathbf{x}, \mathbf{x}')$ that assigns higher similarity scores to more relevant instances, i.e.,

$$S(\mathbf{x}_t, \mathbf{x}_t^+) > S(\mathbf{x}_t, \mathbf{x}_t^-), \forall t \in [T].$$

For example, compared with an image $\mathbf{x}_t^-$, $\mathbf{x}_t^+$ is a more similar image with $\mathbf{x}_t$.

For the similarity function, we adopt a parametric similarity function that has a bi-linear form,

$$S_M(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top M\mathbf{x}',$$

where $M \in \mathbb{R}^{d \times d}$. In order to learn the optimal parameter M, we introduce some loss function $\ell(M; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$ that measures its performance on the t-th triplet. One popular loss function is the well-known hinge loss

$$\ell(M; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-)) = [1 - S_M(\mathbf{x}_t, \mathbf{x}_t^+) + S_M(\mathbf{x}_t, \mathbf{x}_t^-)]_+,$$

where $[\cdot]_+ = \max(0, \cdot)$. The above loss measures how much is the violation of the desired constraint $S_M(\mathbf{x}_t, \mathbf{x}_t^+) \geq S_M(\mathbf{x}_t, \mathbf{x}_t^-)$ by the similarity function defined by M.

Under these settings, an online learning algorithm updates its model in rounds. At each round a new model is estimated, based on the previous one and the current triplet. We denote the

models estimate after the $t-1$-th round by M_t. A typical online algorithm will starts with some initial model, e.g., $M_1 = 0$. On each round t, a triplet $(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-)$ will be presented to the algorithm. The algorithm can update the model from M_t to M_{t+1}, based on the current triplet. The goal of online similarity learning is to minimize the regret of the algorithm which is defined as

$$R_T = \sum_{t=1}^{T} \ell_t(M_t) - \min_M \sum_{t=1}^{T} \ell_t(M) \quad (1)$$

where $\ell_t(M) = \ell(M; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$. If the regret is sublinear, i.e., $R_T \leq o(T)$, it is easy to observe

$$\sum_{t=1}^{T} \ell_t(\bar{M}_T) \leq \min_M \sum_{t=1}^{T} \ell_t(M) + o(T) \quad (2)$$

where $\bar{M}_T = \frac{1}{T}\sum_{t=1}^{T} M_t$, so $\bar{M}_T$ is a good estimate of $\arg\min_M \sum_{t=1}^{T} \ell_t(M)$. In practice, The final outputted model M_T usually will be used for testing. Please note the online triplets are generally not provided in advance.

To solve this problem, some authors proposed an online learning scheme, OASIS in [13], by applying the online Passive Aggressive (PA) learning strategy [34], i.e.,

$$M_{t+1} = \arg\min_M \frac{1}{2}\|M - M_t\|_F^2 + C\ell_t(M),$$

where $C > 0$ is a trade-off parameter for the regularization term and loss. This algorithm enjoys the following closed-form solution

$$M_{t+1} = M_t + \tau_t \mathbf{x}_t (\mathbf{x}_t^+ - \mathbf{x}_t^-)^\top,$$

where $\tau_t = \min(C, \frac{\ell_t(M_t)}{\|\mathbf{x}_t(\mathbf{x}_t^+ - \mathbf{x}_t^-)^\top\|_F^2})$, and $\|\cdot\|_F$ is Frobenius norm. This algorithm is only provided with a mistake bound by the authors. However, a careful analysis should give it a regret bound of order $O(\sqrt{T})$.

Another possible method to solve this problem is based on the Online Gradient Descent (OGD) method [35], i.e.,

$$M_{t+1} = M_t - \eta_t G_t$$

where $G_t = \partial_M \ell_t(M_t)$. The gradient G_t can be computed as follows

$$G_t = \begin{cases} -\mathbf{x}_t(\mathbf{x}_t^+ - \mathbf{x}_t^-)^\top & \text{if } \ell_t(M_t) > 0 \\ 0 & \text{else} \end{cases}$$

This algorithm generally has a regret bound of $O(\sqrt{T})$. However these two algorithms both learn a dense matrix M, which may result in high testing time complexity and memory cost. To solve this issue, we will study sparse online learning techniques to tackle this problem in this paper.

B. Sparse Online Relative Similarity Learning (SORS)

To make the model sparse, we can add sparse regularization function into the online objective function, i.e.,

$$M_{t+1} = \arg\min_M \Big[\ell_t(M) + \lambda r(M) + \frac{\|M - M_t\|_F^2}{2\eta_t}\Big], \quad (3)$$

where $r(M)$ is typically set as

$$r_1(M) = \|M\|_1 = \sum_{ij} |M_{ij}|.$$

This objective function has three terms:

- the first one is to minimize the loss of new similarity matrix on the current triplet;
- the second term is to make the new similarity matrix sparse, where λ is a sparsity parameter. If λ is larger, the model will be sparser.
- the final term is used to keep the new similarity matrix close to the current one.

The reason to use $\|M\|_1$ is that $\|M\|_1$ is a convex upper bound of $\|M\|_0$, which can make the optimal solution sparse, i.e., most of M_{ij} will be zeros. In this way, the computation of $S(\mathbf{x}, \mathbf{x}')$ will be significantly reduced, since $S(\mathbf{x}, \mathbf{x}') = \sum_{ij} M_{ij}\mathbf{x}_i\mathbf{x}'_j$. However, this regularization treats the diagonal elements of M equally with those off diagonal elements, it is clearly insufficient, since diagonal elements should generally be dense while those off-diagonal elements should be sparse. The reason is that the diagonal element represents the self-correlation of a feature while the off-diagonal elements represent correlation between features. Self-correlations are always larger while inter correlations are generally sparse in high dimension space. To solve this issue, an alternative is to adopt the off-diagonal L_1 norm

$$r_2(M) = \|M\|_{1,\text{off}} = \sum_{i \neq j} |M_{ij}|,$$

to pursue a sparse solution while keep the diagonal elements dense.

Given the regularization function, we would solve the online objective function Eq. (3). However this optimization usually does not have a closed-form solution, which will incur additional computational cost. To tackle it, we proposed to linearize the loss function $\ell_t(M)$, to get the following new online objective function

$$\begin{aligned} &M_{t+1} \\ &= \arg\min_M \Big[\ell(M_t) + \langle M - M_t, G_t\rangle + \lambda r(M) + \frac{\|M - M_t\|_F^2}{2\eta_t}\Big] \\ &= \arg\min_M \Big[\langle M, G_t\rangle + \lambda r(M) + \frac{\|M - M_t\|_F^2}{2\eta_t}\Big], \end{aligned} \quad (4)$$

where $\langle M, G_t\rangle = \text{Tr}(M^\top G_t)$ and $G_t = \partial_M \ell_t(M_t)$.

It is easy to verify that this objective will produce the $t+1$-th iterate as

$$M_{t+1} = \text{prox}_{\eta_t \lambda r}(M_t - \eta_t G_t), \quad (5)$$

where $\text{prox}_h(M) = \arg\min_N \Big(h(N) + \frac{1}{2}\|N - M\|_F^2\Big)$. Luckily, when $r_1(M) = \|M\|_1$, we have

$$\text{prox}_{\eta_t \lambda r_1}(M) = \text{sign}(M) \odot [|M| - \eta_t\lambda]_+, \quad (6)$$

and when $r_2(M) = \|M\|_{1,\text{off}}$, we have

$$\text{prox}_{\eta_t \lambda r_2}(M) = \text{sign}(M) \odot [|M| - \eta_t\lambda(1 - I)]_+, \quad (7)$$

where I is the identity matrix. $\odot$ is element-wise product for two matrices. These updates can be computed efficiently.

Finally, we can summarize the proposed Sparse Online Relative Similarity Learning algorithm in the Algorithm 1.

Algorithm 1 **S**parse **O**nline **R**elative **S**imilarity Learning **(SORS)**

Input: Regularization parameter $\lambda > 0$, learning rates $\eta_t > 0$, and regularizer $r(\cdot)$.
Initialize: $M_1 = I$.
for $t = 1, \ldots, T$ **do**
  Receive $(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-)$;
  Compute $\ell(M_t; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$;
  $G_t = \partial_M \ell(M_t; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$;
  $M_{t+1} = \text{prox}_{\eta_t \lambda r}(M_t - \eta_t G_t)$;
end for
Output: M_{T+1}

C. Adaptive Sparse Online Relative Similarity Learning (AdaSORS)

To improve the learning performance of relative similarity learning, a second order similarity learning algorithm AROMA is proposed [36]. AROMA is based on the well-known Confidence Weighted learning strategy [37], which does not only use the first order information, i.e., weighted mean of examples, but also the second-order information, i.e., the covariance matrix for all the features, to update the model. Theoretical results and empirical comparisons have shown that AROMA can significantly outperform OASIS in terms of information retrieval performance. However, the model produced by AROMA is still dense.

In this subsection, we would like to adopt the second order information of the data stream to further improve the learning efficiency and efficacy of sparse online relative similarity learning algorithm. Different from AROMA, we will utilize the idea of AdaGrad [38], [39] to incorporate the second order information through adaptive regularization. Specifically, we will maintain a matrix $\Sigma_t = \delta + H_t$, where δ is a smooth parameter to keep each element of Σ_t invertible and H_t is similar with correlation matrix between features. Specifically, H_1 will be initialized as zero matrix, and at the t-th iteration, it will be updated using the following rule:

$$H_{t,ij} = \sqrt{H_{t-1,ij}^2 + G_{t,ij}^2},$$

$\forall i, j \in [d]$. Given the matrix Σ_t, we can introduce a norm induced by Σ_t, which is defined as follows:

$$\|M\|_{\Sigma_t}^2 = \sum_{ij} \Sigma_{t,ij} M_{ij}^2.$$

Given this norm, we can follow the similar idea of AdaGrad to replace the final term of the first-order objective function of SORS with this new norm, to get the following updating strategy

$$M_{t+1} = \arg\min_M \Big[\langle M, G_t\rangle + \lambda r(M) + \frac{\|M - M_t\|_{\Sigma_t}^2}{2\eta_t}\Big], \quad (8)$$

which is considered as the objective of Adaptive Sparse Online Relative Similarity learning algorithm. The unique difference between this objective and SORS is the final proximity term.

If we define a proximal operator as

$$\text{prox}_h^{\Sigma_t}(M) = \arg\min_N (h(N) + \frac{1}{2}\|N - M\|_{\Sigma_t}^2),$$

then, it is easy to see the solution for AdaSORS is

$$M_{t+1} = \text{prox}_{\eta_t \lambda r}^{\Sigma_t}(M_t - \eta_t G_t ./ \Sigma_t). \quad (9)$$

where $./$ is element-wise division.

When r is set as $r_1 = \|M\|_1$, the corresponding proximal operator is

$$\text{prox}_{\eta_t \lambda r_1}^{\Sigma_t}(M) = \text{sign}(M) \odot [|M| - \lambda \eta_t ./ \Sigma_t]_+, \quad (10)$$

and when r is set as $r_2 = \|M\|_{1,off}$, the corresponding proximal operator is

$$\text{prox}_{\eta_t \lambda r_2}^{\Sigma_t}(M) = \text{sign}(M) \odot [|M| - \lambda \eta_t (1 - I) ./ \Sigma_t]_+, \quad (11)$$

whose's time complexities are $O(d^2)$ so that they can be conducted efficiently.

Finally, we can summarize the proposed Adaptive Sparse Online Relative Similarity Learning algorithm in the Algorithm 2.

Algorithm 2 **Ada**ptive **S**parse **O**nline **R**elative **S**imilarity Learning **(AdaSORS)**

Input: Regularization parameter $\lambda > 0$, learning rates $\eta_t > 0$, smooth parameter δ, and regularizer $r(\cdot)$.
Initialize: $M_1 = I$.
for $t = 1, \ldots, T$ **do**
  Receive $(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-)$;
  Compute $\ell(M_t; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$;
  $G_t = \partial_M \ell(M_t; (\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-))$;
  $H_{t,ij} = \sqrt{H_{t-1,ij}^2 + G_{t,ij}^2}$, $\forall i, j \in [d]$;
  $\Sigma_t = \delta + H_t$;
  $M_{t+1} = \text{prox}_{\eta_t \lambda r}^{\Sigma_t}(M_t - \eta_t G_t ./ \Sigma_t)$;
end for
Output: M_{T+1}

D. Time Complexity Analysis

The proposed algorithms' computational complexities depend on the number of features (d) of an instance. During each online learning step, element-wise operations are involved between $M_t \in \mathbb{R}^{d \times d}$ and $G_t \in \mathbb{R}^{d \times d}$, so the time complexity of SORS is $O(d^2)$ for each learning iteration. AdaSORS needs to deal with matrix Σ_t and H_t, it consumes double time than SORS: $O(2d^2)$. If the instances are sparse, then generally we can compute and update the matrices G, M, Σ, and H more efficiently.

E. Theoretical Analysis

In this subsection, we will provide theoretical upper bounds for the regret of the proposed algorithms which is defined as follows,

$$R_T = \sum_{t=1}^{T} [\ell_t(M_t) + \lambda r(M_t)] - \sum_{t=1}^{T} [\ell_t(M_*) + \lambda r(M_*)],$$

where $M_* = \arg\min_M \sum_{t=1}^{T} [\ell_t(M) + \lambda r(M)]$. This regret is different from the definition in the Eq. (1), since this regret also considers the sparsity regularization term. The bound on the regret implies the gap between the cumulative loss of online learning algorithm and the cumulative loss of the best model which can be chosen in hindsight.

Now, we will analyze the theoretical performance of the proposed algorithms for the case that ℓ_t is Lipschitz continuous e.g., $\ell_t(M) = [1 - \mathbf{x}_t^\top M \mathbf{x}_t^+ + \mathbf{x}_t^\top M \mathbf{x}_t^-]_+$.

Theorem 1. *Let $\{(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-) \mid t = 1, \ldots, T\}$ be a sequence of triplets, where $\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^- \in \mathbb{R}^d$. Suppose $\ell_t(M)$ is convex over $\mathbb{R}^{d\times d}$ and L-Lipschitz for any $t \in [T]$, if the SORS algorithm is run on this sequence of triplets with $\eta_t = \eta$, then we have the following regret bound,*

$$R_T \leq \frac{1}{2\eta}\|M_*\|_F^2 + \frac{\eta}{2}L^2 T.$$

Furthermore, if we set $\eta = \|M_*\|/(L\sqrt{T})$, then for all $T > 0$, we have

$$R_T \leq \|M_*\|_F L\sqrt{T}.$$

Remark: This theorem implies the SORS algorithms will suffer a regret of order $O(\sqrt{T})$ and an average of regret of order $O(1/\sqrt{T})$. If T approaches infinity, the average regret vanishes.

Theorem 2. *Let $\{(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-) \mid t = 1, \ldots, T\}$ be a sequence of triplets, where $\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^- \in \mathbb{R}^d$. Suppose $\ell_t(M)$ is convex over $\mathbb{R}^{d\times d}$ and the gradient of ℓ_t with respect M_t is $\nabla_M \ell_t(M_t) = G_t$, if the AdaSORS algorithm is run on this sequence of triplets, then we have the following regret bound*

$$R_T \leq \frac{1}{2\eta}D_M^2 \sum_{i=1}^{d}\sum_{j=1}^{d}\|G_{1:t,ij}\|_2 + \eta\sum_{i=1}^{d}\sum_{j=1}^{d}\|G_{1:t,ij}\|_2,$$

where $D_M = \max_t \|M_* - M_t\|_\infty$ and $\|G_{1:t,ij}\|_2 = \sqrt{\sum_{t=1}^{T} G_{t,ij}^2}$.

Furthermore, if η is set as $D_\infty/\sqrt{2}$, we have

$$R_T \leq \sqrt{2}D_\infty \sum_{i=1}^{d}\sum_{j=1}^{d}\|G_{1:t,ij}\|_2.$$

Remark: According to [38], this regret is in the same order of the regret by choosing the best fixed Σ for the Eq. (8).

IV. Experiments

In this section, we evaluate the proposed approaches on six public high dimensional benchmark datasets to examine their effectiveness.

A. Compared Algorithms

We compared the following approaches:

- **Euclidean**: The baseline measurement method using the standard Euclidean distance in feature space.
- **LMNN**: Largest Margin Nearest Neighbor method proposed by [5].
- **ITML**: Information Theoretic Metric Learning proposed by [24].
- **LEGO**: LogDet Exact Gradient Online Learning proposed by [12].
- **RDML**: Regularized Distance Metric Learning proposed by [27].
- **AROMA**: Adaptive Regularization for Weight Matrices by [20].
- **OASIS**: It learns a bilinear similarity which is based on online Passive Aggressive algorithm using triplet instances [13].
- **SORS**: The algorithm described above in Algorithm 1. **SORS-I** is the algorithm which adopted the Eq. (6) as prox function in Eq. (5). **SORS-II** is the one which adopted the Eq. (7).
- **AdaSORS**: The algorithm described above in Algorithm 2. **AdaSORS-I** is the algorithm which adopted the Eq. (10) as prox function in Eq. (9). **AdaSORS-II** is the one which adopted the Eq. (11).

B. Experimental Datasets and Setup

To examine the performance, we test all the algorithms on six publicly available high dimensional datasets "Caltech256"[1], "BBC"[2], "Gisette"[3], "Protein", "RCV1", and "Sector" from LIBSVM[4], as shown in Table I. The sparsity means the percentage of empty attribute values.

TABLE I. Statistics of Experimental Datasets.

Data Set	Source	Class	# Feature	# Size	Sparsity (%)
Protein	LIBSVM	3	357	17,766	71.00
Caltech256	Caltech	257	1,000	29,461	96.75
Gisette	UCI	2	5,000	6,000	00.91
BBC	UCD	5	9,636	2,225	98.66
RCV1	LIBSVM	53	47,236	8,967	99.86
Sector	LIBSVM	105	55,197	9,619	99.70

Prediction for the structure and function of proteins plays an important role in bioinformatics. Protein dataset is used to predict the local conformation of the polypeptide chain [40]. This dataset comprises 17,766 protein entries with 3 categories. Each protein entry is represented by 357 features.

Caltech256 is a dataset for image retrieval tasks, which consists of 30,607 images assigned to 257 categories, where each image is represented by 1,000 features. We use the images from the first 10 and 50 classes to construct experimental datasets, which are denoted as "Caltech10" and "Caltech50".

Gisette is a dataset used for a handwritten recognition problem, which is to separate the confusable digits handwritten number '4' and '9' [41]. The Gisette dataset is released from NIPS 2003 Feature Selection Challenge, which contains 6,000 entries. Each protein entry is represented by 5,000 features. From Table I, we can see that the entry in Gisette dataset is dense.

The Reuters RCV1 is a text classification dataset [42]. This dataset contains documents originally written in five different languages over a common set of 6 categories. Each RCV1 entry is represented by 47,236 features. We use the first 10 categories as our study dataset.

BBC news article dataset was gathered from BBC news website, which corresponds to stories in five topical areas from 2004-2005 [43]. The dataset consists of 2,225 documents in five areas: business, entertainment, politics, sport and tech. Each document is represented by 9,636 features.

Sector dataset is used for text categorization [44]. The scaled data consists of company web pages classified in a

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech256

[2] http://mlg.ucd.ie/datasets/bbc.html

[3] http://archive.ics.uci.edu/ml/datasets.html

[4] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets

TABLE II. EVALUATION OF THE LEARNING ALGORITHMS ON THE DATA SETS.

Algorithm	Protein (#Class=3, #Iterations=10^5)			Caltech50 (#Class=50, #Iterations=10^5)		
	MAP (%)	Sparsity (%)	Time (s)	MAP (%)	Sparsity (%)	Time (s)
Eucl.	36.93±0.01	99.9±0.00	0	10.05±0.21	99.9±0.00	0
LMNN	45.91±0.01	0.56±0.01	78088.84±219.14	8.07±1.10	5.91±0.01	106884.06±894.25
ITML	37.13±0.01	0.56±0.01	50.00±0.01	10.29±0.20	6.83±0.57	1124.88±26.01
LEGO	37.44±0.01	0.56±0.01	60.00±0.01	10.15±0.01	5.52±0.01	531.94±4.62
RDML	38.96±0.01	0.58±0.01	108.90±8.20	10.10±0.22	7.50±0.10	623.78±12.13
OASIS	43.91±0.16	0.56±0.01	86.51±15.18	12.93±0.20	5.94±0.10	673.43±89.74
AROMA	44.58±0.22	0.56±0.01	822.93±71.38	13.39±0.10	5.93±0.12	1294.66±147.32
SORS-I	45.66±0.16	1.02±0.04	217.83±0.54	13.28±0.14	13.82±0.14	895.79±27.87
SORS-II	45.69±0.06	0.81±0.03	239.62±1.18	13.29±0.18	13.83±0.18	958.97±20.19
AdaSORS-I	47.02±0.41	0.97±0.05	400.11±9.21	13.82±0.14	14.10±1.21	2552.17±169.52
AdaSORS-II	46.45±0.39	0.99±0.04	423.27±10.91	13.83±0.15	14.10±1.26	2993.64±282.13
Algorithm	Gisette (#Class=2, #Iterations=10^5)			BBC (#Class=5, #Iterations=10^5)		
	MAP (%)	Sparsity (%)	Time (s)	MAP (%)	Sparsity (%)	Time (s)
Eucl.	62.05±0.01	99.9±0.00	0	32.91±0.01	99.9±0.00	0
LMNN	87.01±0.24	1.75±0.01	78994.68±141.21	NA	NA	NA
ITML	62.09±0.32	28.64±0.01	750.82±23.12	31.78±0.47	10.86±1.43	109687.28±633.54
LEGO	78.38±0.36	6.45±0.01	5410.54±30.94	30.29±0.01	3.44±0.26	31827.25±59.46
RDML	82.49±0.32	20.73±0.01	2090.51±7.88	70.79±2.35	83.75±0.01	8189.38±188.18
OASIS	83.49±0.22	4.72±0.01	2316.77±469.77	79.36±0.41	61.62±0.24	8881.26±475.22
AROMA	83.57±1.82	3.89±0.18	26097.59±27.40	79.68±0.43	60.83±1.23	9723.02±381.83
SORS-I	88.89±2.49	4.01±1.62	16839.32±101.85	91.03±1.12	78.67±3.56	90370.71±519.35
SORS-II	89.89±1.22	4.08±1.61	22578.96±1028.02	92.20±1.28	78.41±3.21	135376.85±678.71
AdaSORS-I	90.92±0.71	4.40±1.10	31256.19±1206.14	92.43±0.62	82.15±2.68	303718.57±643.43
AdaSORS-II	90.99±0.53	4.20±1.12	35471.63±848.63	94.09±0.61	81.24±2.35	364976.73±1485.68
Algorithm	RCV1 (#Class=10, #Iterations=10^5)			Sector (#Class=105, #Iterations=10^5)		
	MAP (%)	Sparsity (%)	Time (s)	MAP (%)	Sparsity (%)	Time (s)
Eucl.	58.28±0.01	99.9±0.00	0	19.40±0.01	1±0.00	0
OASIS	92.37±0.01	97.14±0.01	30222.26±792.28	65.18±0.01	86.98±0.01	196561.51±1862.72
AROMA	92.54±0.01	97.11±0.01	40407.61±149.85	65.47±0.01	87.91±0.01	275107.98±4295.58
SORS-I	92.62±0.01	99.84±0.01	71473.62±1370.12	66.22±1.02	98.12±0.01	311849.83±3976.32
SORS-II	92.62±0.01	99.84±0.01	100543.28±4981.78	66.89±1.02	98.12±0.01	381573.25±10874.97
AdaSORS-I	92.82±0.01	99.64±0.01	148260.37±3411.35	67.18±0.01	98.20±0.01	632147.31±21041.49
AdaSORS-II	92.86±0.01	99.64±0.01	162515.32±9431.53	67.10±0.01	98.20±0.01	687947.24±15972.15

hierarchy of industry sectors. It was collected from web sites belonging to industry companies from various economic sectors. The whole dataset contains 9,619 entries. Each Sector entry is represented by 55,197 features.

For each dataset, data instances from each class were randomly split into training set (70%) and test set (30%). To generate a triplet $(\mathbf{x}_t, \mathbf{x}_t^+, \mathbf{x}_t^-)$, $\mathbf{x}_t$ is firstly randomly selected from the whole training set, then $\mathbf{x}_t^+$ is randomly selected from the subset of training set, which consists of the examples with the same class of $\mathbf{x}_t$, finally $\mathbf{x}_t^-$ is randomly selected from the rest of training set, which consists of the examples with different classes of $\mathbf{x}_t$.

To make a fair comparison, all algorithms adopt the same experimental setup. We use cross-validation to select the values of hyper parameters for all algorithms. Specifically, the parameters set by cross validation include: the μ parameter for LMNN ($\mu \in \{0.1,0.2,...0.9\}$), the γ parameter for ITML ($\gamma \in \{0.1,1,10\}$), the regularization parameter η for LEGO ($\eta \in \{0.02,0.08,0.32\}$), the λ parameter for RDML ($\lambda \in \{0.01,0.1\}$), the scalar parameter r for AROMA ($r \in \{0.01,0.1,1,10,20...100\}$) and the aggressiveness parameter C for OASIS ($C \in \{0.1,0.08,0.06,0.04...\}$). All the evaluation results listed below were achieved by choosing the parameters using cross validation.

C. Evaluation Measures

We evaluate the performance of the algorithms, by the sparsity of the model, the precision at top k retrieval results, the mean average precision of retrieval results, and the running time.

- The sparsity of a matrix M is defined as Sparsity$= 1 - \frac{\|M\|_0}{d^2}$, which is controlled by λ in Eq. (4) and Eq. (8).
- Precision at top-k corresponds to the number of relevant results on the first k query results.
- The *mean average precision* (MAP) is used to evaluate the accuracy of the retrieval performance.
- The training time is also reported for each algorithm to show the computational efficiency.

D. Performance Evaluation

Table II summarizes the performance of all the compared similarity/metric learning algorithms over all the benchmark datasets, where results for LMNN on "BBC" "RCV1" and "Sector" are not reported since the learning task can not be finished in 168 hours. From the results, we can draw several observations.

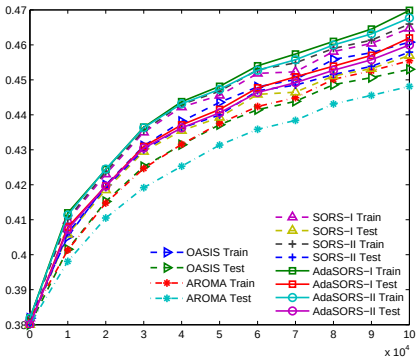
Firstly, the sparsity of Euclidean metric is very high, however its MAP performance is much lower than those of other algorithms for most of cases, which implies the importance of metric learning for improving the information retrieval performance.

Secondly, among the baseline algorithms in comparison, we observe that the offline algorithm "LMNN" is generally significantly slower than the other online learning algorithms including, ITML, LEGO, RDML, and OASIS, although it generally achieve better MAP than the online learning algorithms. This verified the efficiency and scalability of online learning, which makes them more attractive for large-scale tasks and mining data streams.

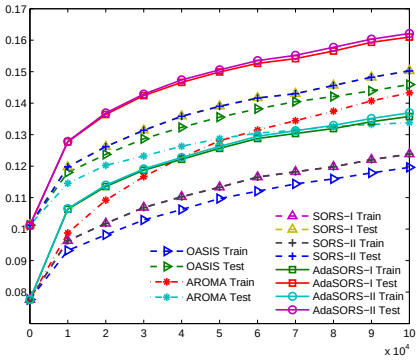
Thirdly, among the online learning algorithms in comparison, we observe that the online similarity learning algorithms (OASIS and AROMA) generally achieve better or at least comparable MAP performance compared with online metric learning algorithms (ITML, LEGO, RDML), which indicates the effectiveness of similarity learning techniques, especially that learning similarity function without PSD constraint is effective. Compared with OASIS, AROMA generally achieves better MAP performance on all the datasets, which verified the effectiveness of introducing second order information to improve the learning efficacy

Finally, comparing with all the baseline algorithms, the proposed SORS algorithms (SORS, AdaSORS) achieve the highest test MAP performance on all the datasets. This shows that the proposed learning strategy is effective in improve the generalization ability of the learnt model. Secondly, by examining the sparsity of resulting matrix, we observe that SORS algorithms result in the sparsest model than the other baseline algorithms for most of the datasets (Although ITML produces sparser model for Gisette and RDML produces sparser models for Gisetter and BBC, their MAP performance is significantly worse). This implies that the proposed strategy can effectively improve the sparsity of the modal. In addition, compared with SORS-I and SORS-II, AdaSORS algorithms (AdaSORS-I and AdaSORS-II) generally achieve better or comparable MAP, and produce comparably or sparser models.

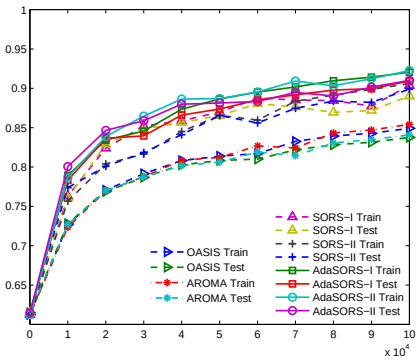
Moreover, Figure 1 shows the MAP performance of all the online similarity learning algorithms in comparison over trials. The parameters for these algorithms are selected by cross validation. Specifically, on the dataset "Protein", the parameters are set as: $C = 0.01$ for OASIS, $r = 60$ for AROAM, $\lambda = 1.e-6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.e-4$, $\eta = 0.1$, $\delta = 5$ for AdaSORS-I & AdaSORS-II . On the dataset "Caltech50", the parameters are set as: $C = 0.06$ for OASIS, $r = 13$ for AROMA, $\lambda = 1.e-6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.e-6$, $\eta = 0.1$, $\delta = 0.1$ for AdaSORS-I & AdaSORS-II. On the dataset "Gisette", the parameters are set as $C = 0.01$ for OASIS, $r = 90$ for AROMA, $\lambda = 1.e-6$, $\eta = 0.06$ for SORS-I & SORS-II, and $\lambda = 1.e-5$, $\eta = 0.06$, $\delta = 20$ for AdaSORS-I & AdaSORS-II. On the dataset "BBC", the parameters are set as: $C = 0.01$ for OASIS, $r = 90$ for AROMA, $\lambda = 1.e-6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.e-6$, $\eta = 0.1$, $\delta = 0.1$ for AdaSORS-I & AdaSORS-II. If not stated, the parameters for the rest of the figures are set as the same as here.



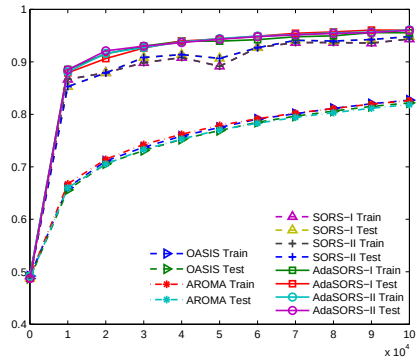
(a) Protein



(b) Caltech50

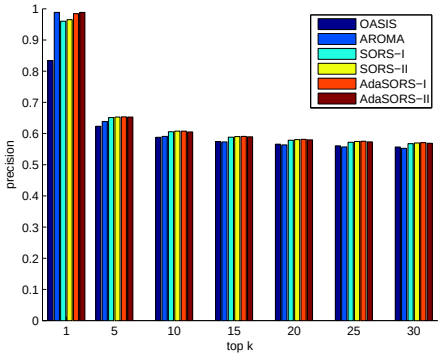


(c) Gisette

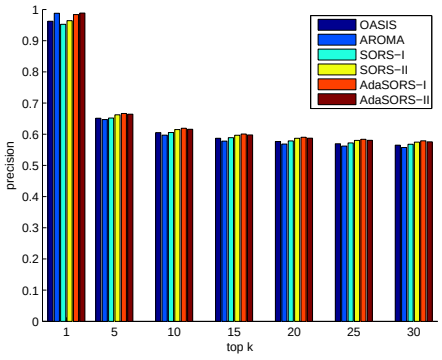


(d) BBC

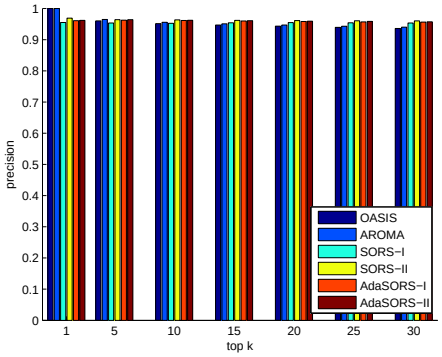
Fig. 1. MAP of online similarity learning algorithms as a function of the number of training iteration on four data sets.



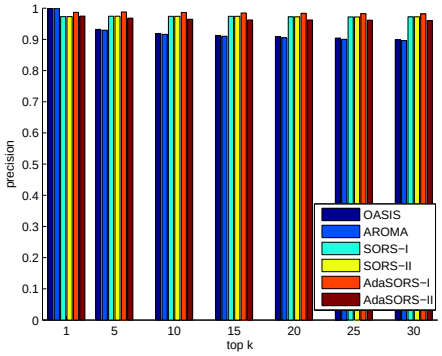
(a) Protein



(b) Caltech50

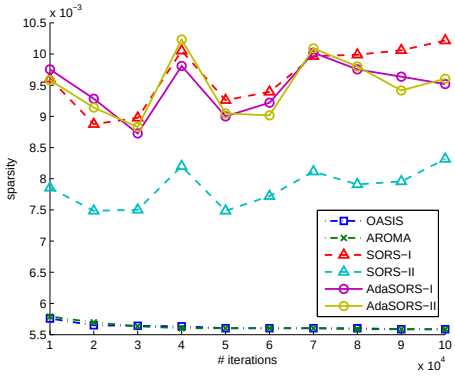


(c) Gisette

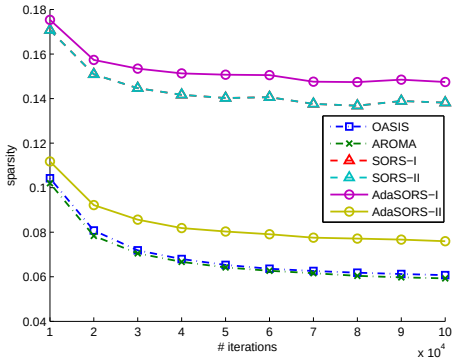


(d) BBC

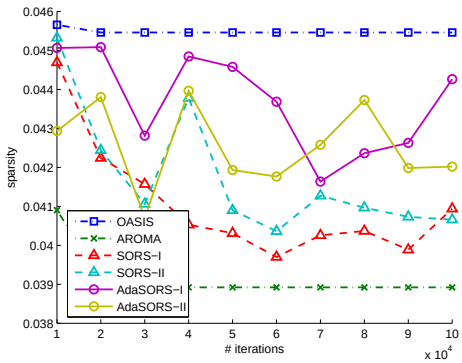
Fig. 2. Precision@k of the online similarity learning algorithms on four data sets.



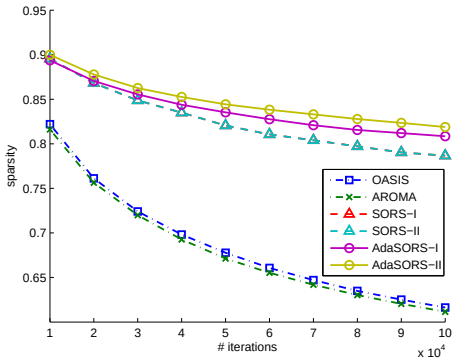
(a) Protein



(b) Caltech50



(c) Gisette



(d) BBC

Fig. 3. Online sparsity of the online similarity learning algorithms on four data sets.

From Figure 1, we can observe that generally SORS algorithms outperform OASIS and AROMA, which indicates the effectiveness of introduction of sparsity. In addition, AdaSORS algorithms generally can achieve the best MAP performance, which verified the effectiveness of using second order information for improving learning efficacy. Overall, these observations are consistent with Table II, which again verified that the introduction of sparse regularization can efficiently remove noisy interactions between features and improve the generalization ability of the learning algorithms.

To further demonstrate the effectiveness of the proposed algorithms, Figure 2 summarizes the precision@k performance of the online similarity learning algorithms: OASIS, AOMA, SORS-I, SORS-II, AdaSORS-I and AdaSORS-II on four datasets. It can be observed that AdaSORS and SORS algorithms overall outperform OASIS and AROMA on most of k (except k=1). The reason for this is the fact SORS and AdaSORS algorithms are designed to optimize the AUC performance, which may not guarantee the best precision at top k. However, the proposed algorithms still achieve better precision at top k for most of cases. All these observations verified the effectiveness of the proposed algorithms.

Moreover, we also estimate the online sparsity of the models produced by online similarity learning algorithms with respect to iterations on four datasets, illustrated in Figure 3. We can observe the sparsity generally decreases as the number of iteration increase for all the algorithms, which is due to the fact more and more rank-1 matrices are added to the model. In addition, AdaSORS and SORS algorithms generally achieve much higher sparsity than AROMA and OASIS (except for Gisette, where all the algorithms achieve comparable sparsity) and AdaSORS algorithms overall produce sparser models than SORS, which again verified the effectiveness of the sparse regularization to reduce memory cost.

E. Testing Efficiency

To evaluate the testing efficiency, Figure 4 summarizes the query time of the models learnt using OASIS, AROMA, SORS-I, SORS-II, AdaSORS-I, and AdaSORS-II on "BBC" for test datasets with varied sizes, for which the sparsity of these models are already shown in the Table II.

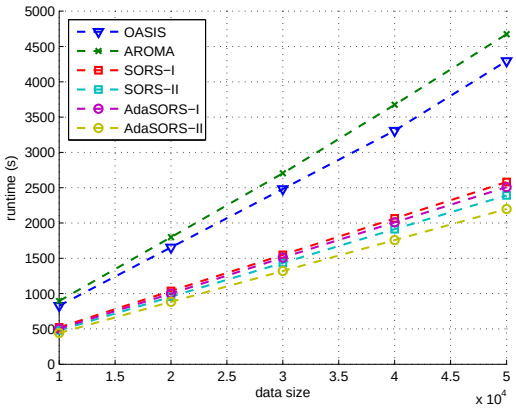


Fig. 4. Comparison of the testing efficiencies of the models from 6 online similarity learning algorithms on "BBC".

Firstly, we can observe that AdaSORS and SORS algorithms generally are more efficient than OASIS and AROMA over all the test sets of different sizes, since the models of these four algorithms are much sparser than OASIS and AROMA. In addition, when the number of queries is small, the time cost of non-sparse algorithms (OASIS, AROMA) and sparse algorithms (SORS-I, SORS-II, AdaSORS-I, AdaSROS-II) are not much different; however if the number of queries is very large, sparse algorithms generally are significantly efficient than non-sparse algorithms, e.g., AdaSORS-II method saved 50% of test time when the size of data is 50K. This makes the proposed algorithms more attractive for large-scale high dimensional real-world applications.

F. Sparsity-MAP Tradeoffs

Finally, we evaluate the relationship between the MAP performance and the sparsity of the models of the online similarity learning algorithms on "Caltech10", which is demonstrated in Figure 5, where the sparsity and MAP of OASIS are constant values.

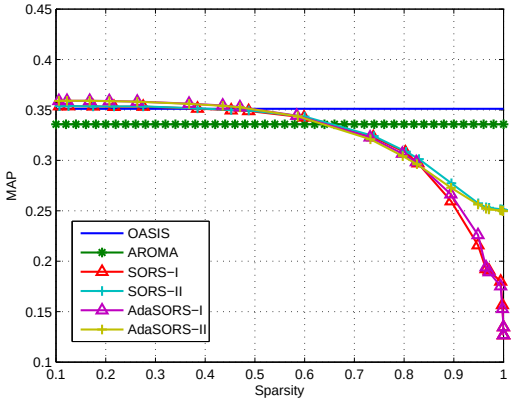


Fig. 5. Tradeoff between MAP and Sparsity of models of online similarity learning algorithms on "Caltech10" dataset for 100K iterations.

Firstly, we can observe that when the sparsity is small enough (less than 0.5), the MAPs performance of the proposed SORS and AdaSORS algorithms tend to be highest and stable, while when the sparsity is larger, increasing the sparsity generally leads to rapid deceasing of MAPs. However, the proposed SORS and AdaSORS algorithms allow us to choose a much sparser model, e.g. with around 15 percentage sparsity, with better or at least comparable MAP performance compared with OASIS and AROMA. In addition, when the sparsity is extremely high, SORS-II and AdaSORS-II can still achieve much better MAP performance than SORS-I and AdaSORS-I respectively, which indicates the importance of those diagonal elements for learning an effective similarity function.

V. Conclusion

This paper presented a Sparse Online Relative Similarity Learning (SORS) framework, and four scalable algorithms for similarity learning tasks with high dimensional features. This framework would like to learn a sparse model which can assign similar items with higher similar values. We theoretically analyzed the regret bounds of the proposed algorithms, and conducted a set of experiments by comparing with a number of

competing algorithms. Promising empirical results shows that the proposed algorithms can effectively learn sparser model with better or at least comparable MAP performance.

ACKNOWLEDGMENT

The work is partially supported by the NSFC (61472149, 71401128), the Fundamental Research Funds for the Central Universities (2015QN67), and the SRF for ROCS, SEM.

REFERENCES

[1] A. Hotho, A. Nürnberger, and G. Paass, "A brief survey of text mining," *LDV Forum*, vol. 20, no. 1, pp. 19–62, 2005.

[2] S. C. H. Hoi, W. Liu, M. R. Lyu, and W. Ma, "Learning distance metrics with contextual constraints for image retrieval," in *CVPR, 17-22 June 2006, New York, NY, USA*, 2006, pp. 2072–2078.

[3] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell, "Distance metric learning with application to clustering with side-information," in *Advances in Neural Information Processing Systems 15 [Neural Information Processing Systems, NIPS 2002, December 9-14, 2002, Vancouver, British Columbia, Canada]*, 2002, pp. 505–512.

[4] J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov, "Neighbourhood components analysis," in *NIPS, December 13-18, 2004, Vancouver, British Columbia, Canada*, 2004, pp. 513–520.

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

[6] B. McFee and G. R. G. Lanckriet, "Metric learning to rank," in *ICML, June 21-24, 2010, Haifa, Israel*, 2010, pp. 775–782.

[7] P. Zhao and S. C. H. Hoi, "OTL: A framework of online transfer learning," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, 2010, pp. 1231–1238.

[8] P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang, "Online AUC maximization," in *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, 2011, pp. 233–240.

[9] P. Zhao, S. C. H. Hoi, and R. Jin, "Double updating online learning," *JMLR*, vol. 12, pp. 1587–1615, 2011.

[10] S. C. H. Hoi, J. Wang, and P. Zhao, "LIBOL: a library for online learning algorithms," *JMLR*, vol. 15, no. 1, pp. 495–499, 2014.

[11] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," *Chemometrics and intelligent laboratory systems*, vol. 50, no. 1, pp. 1–18, 2000.

[12] P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman, "Online metric learning and fast similarity search," in *NIPS*, 2008, pp. 761–768.

[13] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," *JMLR*, vol. 11, pp. 1109–1135, 2010.

[14] P. Wu, Y. Ding, P. Zhao, C. Miao, and S. C. H. Hoi, "Learning relative similarity by stochastic dual coordinate ascent," in *AAAI, July 27 -31, 2014, Québec City, Québec, Canada.*, 2014, pp. 2142–2148.

[15] A. M. Qamar and É. Gaussier, "Online and batch learning of generalized cosine similarities," in *ICDM, Miami, Florida, USA, 6-9 December 2009*, 2009, pp. 926–931.

[16] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *CoRR*, vol. abs/1306.6709, 2013.

[17] P. Wu, S. C. H. Hoi, H. Xia, P. Zhao, D. Wang, and C. Miao, "Online multimodal deep similarity learning with application to image retrieval," in *ACM Multimedia Conference, MM '13, Barcelona, Spain, October 21-25, 2013*, 2013, pp. 153–162.

[18] H. Xia, S. C. H. Hoi, R. Jin, and P. Zhao, "Online multiple kernel similarity learning for visual search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 536–549, 2014.

[19] A. Bellet, A. Habrard, and M. Sebban, "Similarity learning for provably accurate sparse linear classification," in *ICML, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[20] K. Crammer and G. Chechik, "Adaptive regularization for weight matrices," in *ICML, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. ACM, 2012.

[21] G. Qi, J. Tang, Z. Zha, T. Chua, and H. Zhang, "An efficient sparse metric learning in high-dimensional space via l_1-penalized log-determinant regularization," in *ICML*, 2009, p. 106.

[22] Y. Ying, K. Huang, and C. Campbell, "Sparse metric learning via smooth optimization," in *NIPS*, 2009, pp. 2214–2222.

[23] K. Huang, Y. Ying, and C. Campbell, "Generalized sparse metric learning with relative comparisons," *Knowl. Inf. Syst.*, vol. 28, no. 1, pp. 25–45, 2011.

[24] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *ICML*. ACM, 2007, pp. 209–216.

[25] G. Kunapuli and J. W. Shavlik, "Mirror descent for metric learning: A unified approach," in *ECML*, 2012, pp. 859–874.

[26] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng, "Online and batch learning of pseudo-metrics," in *ICML*, 2004.

[27] R. Jin, S. Wang, and Y. Zhou, "Regularized distance metric learning: Theory and algorithm," in *NIPS*, 2009, pp. 862–870.

[28] J. Langford, L. Li, and T. Zhang, "Sparse online learning via truncated gradient," *JMLR*, vol. 10, pp. 777–801, 2009.

[29] J. C. Duchi and Y. Singer, "Efficient online and batch learning using forward backward splitting," *JMLR*, vol. 10, pp. 2899–2934, 2009.

[30] D. Wang, P. Wu, P. Zhao, M. Chunyan, and H. Steven C.H., "High-dimensional data stream classification via sparse online learning," in *14th IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, 2014.

[31] Y. Nesterov, "Primal-dual subgradient methods for convex problems," *Math. Program.*, vol. 120, no. 1, pp. 221–259, 2009.

[32] L. Xiao, "Dual averaging methods for regularized stochastic learning and online optimization," *JMLR*, vol. 11, pp. 2543–2596, 2010.

[33] S. Lee and S. J. Wright, "Manifold identification in dual averaging for regularized stochastic online learning," *JMLR*, vol. 13, pp. 1705–1744, 2012.

[34] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *JMLR*, vol. 7, pp. 551–585, 2006.

[35] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *ICML*, 2003, pp. 928–936.

[36] K. Crammer and G. Chechik, "Adaptive regularization for similarity measures," in *ICML*, 2012.

[37] J. Wang, P. Zhao, and S. C. H. Hoi, "Exact soft confidence-weighted learning," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[38] J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *JMLR*, vol. 12, pp. 2121–2159, 2011.

[39] Y. Ding, P. Zhao, S. C. H. Hoi, and Y. Ong, "An adaptive gradient method for online AUC maximization," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, 2015, pp. 2568–2574.

[40] Z. R. Yang, "Biological applications of support vector machines," *Briefings in Bioinformatics*, vol. 5, no. 4, pp. 328–338, 2004.

[41] I. Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the NIPS 2003 feature selection challenge," in *NIPS, December 13-18, 2004, Vancouver, British Columbia, Canada]*, 2004, pp. 545–552.

[42] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *JMLR*, vol. 5, pp. 361–397, 2004.

[43] D. Greene and P. Cunningham, "Practical solutions to the problem of diagonal dominance in kernel document clustering," in *ICML, Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, 2006, pp. 377–384.

[44] A. McCallum, K. Nigam *et al.*, "A comparison of event models for naive bayes text classification," in *AAAI-98 workshop on learning for text categorization*, vol. 752. Citeseer, 1998, pp. 41–48.