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 instance. 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.

Keywords—High-dimensional similarity learning; Sparse Online learning; Data stream mining;

I. INTRODUCTION

In many applications, it is important to have a suitable similarity measure between objectives, such as text mining [1], image retrieval [2]. 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 learn 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 [13], [14], [15], [16], [17], [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 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 Mahalanobis matrix usually can be as close to the identity matrix as possible [21], so that the learnt Mahalanobis 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 propose 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 propose 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. Section III presents our proposed sparse online relative similarity learning approaches. Section IV conducts an extensive set of experiments by comparison with several state-of-the-art methods, and section V concludes our paper.

II. RELATED WORK

A. Similarity/Metric Learning

Online similarity learning is a group of efficient and scalable machine learning algorithms [10], [14], [25]. Most existing works in similarity learning rely 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 metrics. LMMN [5] addresses the distance learning problem by exploring a large margin nearest neighbor classifier LEGO [12]. LogDet Exact Gradient online, 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 $x_1$ and $x_2$ by $(x_1 - x_2)^\top M(x_1 - x_2)$, 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 $x_1$ and $x_2$, it measures similarity through $x_1^\top M x_2$. 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 [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 $\ell_1$ 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 scheme [28], Truncated Gradient, is proposed. When there are less than a predefined threshold $\theta$, 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 $\{(x_t, x_t^+, x_t^-) \in \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d | t \in [T]\}$ with $[T] = \{1, \ldots, T\}$, where the relevance between $x_t$ and $x_t^+$ is greater than that between $x_t$ and $x_t^-$. Specifically, we would like to learn a similarity function $S(x, x')$ that assigns higher similarity scores to more relevant instances, i.e.,

$$S(x_t, x_t^+) > S(x_t, x_t^-), \forall t \in [T]$$

For example, compared with an image $x_t^-$, $x_t^+$ is a more similar image with $x_t$.

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

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

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

$$\ell(M; (x_t, x_t^+, x_t^-)) = [1 - S_M(x_t, x_t^+) + S_M(x_t, x_t^-)]_+$$

where $[.]_+ = \max(0, \cdot)$. The above loss measures how much the violation of the desired constraint $S_M(x_t, x_t^+) \geq S_M(x_t, x_t^-)$ is 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 starts with some initial models, e.g., $M_1 = 0$. For each round $t$, a triplet $(x_i, x^+_i, x^-_i)$ 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 defined as

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

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

$$\sum_{t=1}^{T} \ell_t(M_T) \leq \min_{M} \sum_{t=1}^{T} \ell_t(M) + o(T)$$

where $M_T = \frac{1}{T} \sum_{t=1}^{T} M_t$, so $M_T$ 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.,

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

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

$$M_{t+1} = M_t + \tau_t x_i (x^+_i - x^-_i)^T$$

where $\tau_t = \min(C, \frac{\ell_t(M_t)}{\|x_i (x^+_i - x^-_i)\|^2_F})$, 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

$$G_t = \begin{cases} 
- x_i (x^+_i - x^-_i)^T & \text{if } \ell_t(M_t) > 0 \\
0 & \text{else}
\end{cases}$$

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.,

$$M_{t+1} = \arg\min_{M} \left[ \ell_t(M) + \lambda r(M) + \frac{\|M - M_t\|^2_F}{2\eta_t} \right]$$

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 $\lambda$ is a sparsity parameter. If $\lambda$ 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 $S(x, x')$ will be significantly reduced, since $S(x, x') = \sum_{ij} M_{ij} x_i x'_j$. However, this regularization treats the diagonal elements of $M$ equally with those off-diagonals, it is clearly insufficient, since diagonal elements should generally be dense while those off-diagonals 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

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

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 propose to linearize the loss function $\ell_t(M)$, to get the following new online objective function

$$M_{t+1} = \arg\min_{M} \left[ \ell(M_t) + (M - M_t, G_t) + \lambda r(M) + \frac{\|M - M_t\|^2_F}{2\eta_t} \right]$$

where $(M, G_t) = \text{Tr}(M^T G_t)$ 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

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

where $\text{prox}_{\eta_t \lambda} r(M) = \arg\min_{N} \left( h(N) + \frac{1}{2\eta_t} \|N - M\|^2_F \right)$. Luckily, when $r_1(M) = \|M\|_1$, we have

$$\text{prox}_{\eta_t \lambda r_1} r_1 = \text{sign}(M) \odot (|M| - \eta_t \lambda (1 - I))_+$$

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

$$\text{prox}_{\eta_t \lambda r_2} r_2(M) = \text{sign}(M) \odot (|M| - \eta_t \lambda (1 - I))_+$$
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 Algorithm 1.

**Algorithm 1 Sparse Online Relative Similarity Learning (SORS)**

1. **Input:** Regularization parameter \( \lambda > 0 \), learning rates \( \eta_i > 0 \), and regularizer \( r(\cdot) \).
2. **Initialize:** \( M_i = I \).
3. **for** \( t = 1, \ldots, T \) **do**
   4. Receive \((x_i, x_i^+, x_i^-)\);
   5. Compute \(\ell(M_t; (x_i, x_i^+, x_i^-))\);
   6. \(G_t = \partial M_t \ell(M_t; (x_i, x_i^+, x_i^-))\);
   7. \(M_{t+1} = \text{prox}_{\eta_t, \lambda r}(M_t - \eta_t G_t)\);
5. **end for**
6. **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 OAS 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 \( \delta \) is a smooth parameter to keep each element of \( \Sigma_t \) 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 \( \Sigma_t \), we can introduce a norm induced by \( \Sigma_t \), which is defined as follow:

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

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

\[
M_{t+1} = \arg \min_M \left\{ \ell(M, G_t) + \lambda r(M) + \frac{\| M - M_t \|_{\Sigma_t}^2}{2\eta_t} \right\} \tag{8}
\]

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

\[
\text{prox}_{\eta_t, \lambda r}(M) = \arg \min_N (h(N) + \frac{1}{2\eta} \| N - M \|_{\Sigma_t}^2)
\]

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

\[
M_{t+1} = \text{prox}_{\eta_t, \lambda r}(M_t - \eta_t G_t) \tag{9}
\]

where \( ./ \) is element-wise division.

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

\[
\text{prox}_{\eta_t, \lambda r_1}(M) = \text{sign}(M) \circ \| M \| - \lambda \eta_t / \Sigma_t \tag{10}
\]

and when \( r \) is set as \( r_2 = \| M \|_{1,0, f} \), the corresponding proximal operator is

\[
\text{prox}_{\eta_t, \lambda r_2}(M) = \text{sign}(M) \circ \| M \| - \lambda \eta_t (1 - 1) / \Sigma_t \tag{11}
\]

whose 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 Algorithm 2.

**Algorithm 2 Adaptive Sparse Online Relative Similarity Learning (AdaSORS)**

1. **Input:** Regularization parameter \( \lambda > 0 \), learning rates \( \eta_i > 0 \), smooth parameter \( \delta \), and regularizer \( r(\cdot) \).
2. **Initialize:** \( M_1 = I \).
3. **for** \( t = 1, \ldots, T \) **do**
   4. Receive \((x_i, x_i^+, x_i^-)\);
   5. Compute \(\ell(M_t; (x_i, x_i^+, x_i^-))\);
   6. \(G_t = \partial M_t \ell(M_t; (x_i, x_i^+, x_i^-))\);
   7. \(H_{t,ij} = \sqrt{H_{t-1,ij}^2 + G_{t,ij}^2}, \forall i, j \in [d]\);
   8. \(\Sigma_t = \delta + H_t\);
   9. \(M_{t+1} = \text{prox}_{\eta_t, \lambda r}(M_t - \eta_t G_t / \Sigma_t)\);
5. **end for**
6. **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 \( \Sigma_t \) 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 \( G, M, \Sigma, \) and \( H \) more efficiently.

**E. Theoretical Analysis**

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

\[
R_T = \sum_{t=1}^{T} (\ell_t(M_t) + \lambda r(M_t)) - \sum_{t=1}^{T} (\ell_t(M_0) + \lambda r(M_0))
\]
where \( M_\ast = \arg\min_M \sum_{t=1}^T [\ell_t(M) + \lambda r(M)] \). 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 \( \ell_t \) is Lipschitz continuous e.g., \( \ell_t(M) = [1 - x_t^\top My_t + x_t^\top My_t^\ast]_+ \).

**Theorem 1.** Let \( \{(x_t, x_t^+, x_t^-) \mid t = 1, \ldots, T\} \) be a sequence of triplets, where \( x_t, x_t^+, x_t^- \in \mathbb{R}^d \). 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,

\[
R_T \leq \frac{1}{2\eta} \| M_\ast \|^2_F + \frac{\eta}{2} L^2 T
\]

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

\[
R_T \leq \frac{\| M_\ast \|^2_F}{T} L \sqrt{T}
\]

**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 \( \{(x_t, x_t^+, x_t^-) \mid t = 1, \ldots, T\} \) be a sequence of triplets, where \( x_t, x_t^+, x_t^- \in \mathbb{R}^d \). Suppose \( \ell_t(M) \) is convex over \( \mathbb{R}^{d \times d} \) and the gradient of \( \ell_t \) 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 \( \frac{D_M}{\sqrt{T}} = \max_t \| M_\ast - M_t \|_\infty \) and \( \| G_{1:t,ij} \|_2 = \sqrt{\sum_{i=1}^T G_{t,ij}^2} \).

Furthermore, if \( \eta \) is set as \( D_\infty / \sqrt{T} \), we have

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

**Remark:** According to [38], this regret is in the same order of the regret by choosing the best fixed \( \Sigma \) 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 compare 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 proposed by [20].
- **OASIS:** It learns a bilinear similarity based on online Passive Aggressive algorithm using triplet instances [13].
- **SORS:** The algorithm described above in Algorithm 1. **SORS-I** is the algorithm adopted the Eq. (6) as prox function in Eq. (5). **SORS-II** is the one adopted the Eq. (7).
- **AdaSORS:** The algorithm described above in Algorithm 2. **AdaSORS-I** is the algorithm adopted the Eq. (10) as prox function in Eq. (9). **AdaSORS-II** is the one 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 in Table I means the percentage of empty attribute values.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Source</th>
<th>Class</th>
<th>Feature</th>
<th>Size</th>
<th>Sparsity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protein</td>
<td>LIBSVM</td>
<td>3</td>
<td>357</td>
<td>17,766</td>
<td>71.00</td>
</tr>
<tr>
<td>Caltech256</td>
<td>Caltech</td>
<td>257</td>
<td>1,000</td>
<td>29,461</td>
<td>96.75</td>
</tr>
<tr>
<td>Gisette</td>
<td>UCI</td>
<td>2</td>
<td>5,000</td>
<td>6,000</td>
<td>90.91</td>
</tr>
<tr>
<td>BBC</td>
<td></td>
<td>5</td>
<td>9,636</td>
<td>2,225</td>
<td>98.06</td>
</tr>
<tr>
<td>RCV1</td>
<td>LIBSVM</td>
<td>53</td>
<td>47,236</td>
<td>8,967</td>
<td>99.86</td>
</tr>
<tr>
<td>Sector</td>
<td>LIBSVM</td>
<td>105</td>
<td>55,197</td>
<td>9,619</td>
<td>99.70</td>
</tr>
</tbody>
</table>

[^4]: [http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets]
entry is represented by 47,236 features. We use the first 10 categories as our study dataset.

Sector dataset is used for text categorization [44]. The scaled data consists of company web pages classified as a hierarchy of industry sectors. It is 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 are randomly split into training set (70%) and test set (30%). To generate a triplet $(x_t, x_t^+, x_t^-)$, $x_t$ is firstly randomly selected from the whole training set, then $x_t^+$ is randomly selected from the subset of training set, which consists of the examples with the same class of $x_t$, finally $x_t^-$ is randomly selected from the rest of training set, which consists of the examples with different classes of $x_t$.

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 $\mu$ parameter for LMNN ($\mu \in \{0.1, 0.2, \ldots, 0.9\}$), the $\gamma$ parameter for ITML ($\gamma \in \{0.1, 1.1\}$), the regularization parameter $\eta$ for LEGO ($\eta \in \{0.02, 0.08, 0.32\}$), the $\lambda$ parameter for RDML ($\lambda \in \{0.01, 0.1\}$), the scalar parameter $r$ for AROMA ($r \in \{0.01, 0.1, 1.1, 10, 20, \ldots, 100\}$) and the aggressiveness parameter $C$ for OASIS ($C \in \{0.1, 0.08, 0.06, 0.04, \ldots\}$).

### 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 $\text{Sparsity} = 1 - \frac{\|M\|_0}{|M|}$, which is controlled by $\lambda$ 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.

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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Caltech50 (#Class=50, #Iterations=10^3)</th>
<th>AdaSORS-II</th>
<th>ADA SORS-I</th>
<th>SORS-II</th>
<th>SORS-I</th>
<th>RDM</th>
<th>AROMA</th>
<th>Eucl.</th>
<th>Time (s)</th>
<th>Sparsity (%)</th>
<th>MAP (%)</th>
<th>Time (s)</th>
<th>Sparsity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdaSORS-II</td>
<td>47.02±0.41</td>
<td>0.97±0.05</td>
<td>400.11±9.21</td>
<td>13.82±0.14</td>
<td>0.14±0.12</td>
<td>2552.17±169.52</td>
<td>13.83±0.15</td>
<td>10.14±0.26</td>
<td>2993.64±282.13</td>
<td>10.05±0.21</td>
<td>99.9±0.00</td>
<td>0</td>
<td>10.05±0.21</td>
</tr>
</tbody>
</table>
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 achieves better MAP than the online learning algorithms. This verifies 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 verifies 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 improving 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 Gisette 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, $\lambda = 1.0e - 6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.0e - 4$, $\eta = 0.1$, $\delta = 5$ for AdaSORS-I & AdaSORS-II. On the dataset “Caltech50”, the parameters are set as: $C = 0.06$ for OASIS, $r = 13$ for AROAM, $\lambda = 1.0e - 6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.0e - 6$, $\eta = 0.1$, $\delta = 0.1$ for AdaSORS-I & AdaSORS-II. On the dataset “Gisette”, the parameters are set as $C = 0.01$ for OASIS, $r = 90$ for AROAM, $\lambda = 1.0e - 6$, $\eta = 0.06$ for SORS-I & SORS-II, and $\lambda = 1.0e - 5$, $\eta = 0.06$, $\delta = 20$ for AdaSORS-I & AdaSORS-II. On the dataset “BBC”, the parameters are set as: $C = 0.01$ for OASIS, $r = 90$ for AROAM, $\lambda = 1.0e - 6$, $\eta = 0.1$ for SORS-I & SORS-II, and $\lambda = 1.0e - 6$, $\eta = 0.1$. 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 verify the effectiveness of the proposed algorithms.

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 achieves better MAP than the online learning algorithms. This verifies the efficiency and scalability of online learning, which makes them more attractive for large-scale tasks and mining data streams.

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 increases 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 verifies 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.

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

Fig. 2. Precision@k of the online similarity learning algorithms on four data sets.
Fig. 3. Online sparsity of the online similarity learning algorithms on four data sets.

(a) Protein

(b) Caltech50

(c) Gisette

(d) BBC

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

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.

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 analyze the regret bounds of the proposed algorithms, and conduct 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 NSFC (No.61472149, No.71401128), the Fundamental Research Funds for the Central Universities (2015QN67), and the SRF for ROCS, SEM. The authors would like to thank Zhenzhong Lan for the fruitful comments on this work.

REFERENCES