Efficient Multiple Instance Metric Learning using Weakly Supervised Data

Marc T. Law[1] Yaoliang Yu[2] Raquel Urtasun[1] Richard S. Zemel[1] Eric P. Xing[3]

[1]University of Toronto [2]University of Waterloo [3]Carnegie Mellon University

Abstract

We consider learning a distance metric in a weakly supervised setting where "bags" (or sets) of instances are labeled with "bags" of labels. A general approach is to formulate the problem as a Multiple Instance Learning (MIL) problem where the metric is learned so that the distances between instances inferred to be similar are smaller than the distances between instances inferred to be dissimilar. Classic approaches alternate the optimization over the learned metric and the assignment of similar instances. In this paper, we propose an efficient method that jointly learns the metric and the assignment of instances. In particular, our model is learned by solving an extension of kmeans *for MIL problems where instances are assigned to categories depending on annotations provided at bag-level. Our learning algorithm is much faster than existing metric learning methods for MIL problems and obtains state-of-the-art recognition performance in automated image annotation and instance classification for face identification.*

1. Introduction

Distance metric learning [33] aims at learning a distance metric that satisfies some similarity relationships among objects in the training dataset. Depending on the context and the application task, the distance metric may be learned to get similar objects closer to each other than dissimilar objects [20, 33], to optimize some k nearest neighbor criterion [31] or to organize similar objects into the same clusters [15, 18]. Classic metric learning approaches [15, 16, 17, 18, 20, 31, 33] usually consider that each object is represented by a single feature vector. In the face identification task, for instance, an object is the vector representation of an image containing one face; two images are considered similar if they represent the same person, and dissimilar otherwise.

Although these approaches are appropriate when each example of the dataset represents only one label, many visual benchmarks such as *Labeled Yahoo! News* [2], UCI Corel5K [7] and Pascal VOC [8] contain images that include multiple labels. We focus in this paper on such multi-label contexts which may differ significantly. In particular, the way in which labels are provided differs in the applications that we consider. To facilitate the presentation, Fig. 1 illustrates an example of the *Labeled Yahoo! News* dataset: the item is a document which contains one image representing four celebrities. Their presence in the image is extracted by a text detector applied on the caption related to the image in the document; the labels extracted from text indicate the presence of several persons in the image but do not indicate their exact locations, *i.e.*, the correspondence between the labels and the faces in the image is unknown. In the Corel5K dataset, image labels are tags (*e.g.*, *water*, *sky*, *tree*, *people*) provided at the image level.

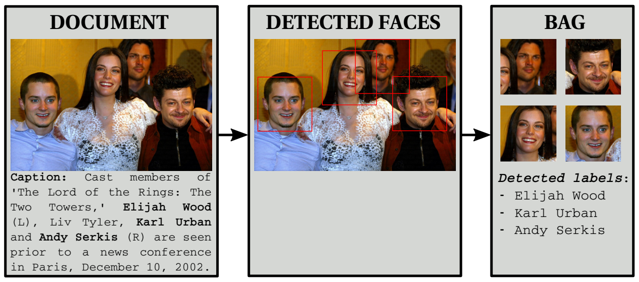
DOCUMENT
Caption: Cast members of 'The Lord of the Rings: The Two Towers,' **Elijah Wood** (L), Liv Tyler, **Karl Urban** and **Andy Serkis** (R) are seen prior to a news conference in Paris, December 10, 2002.
DETECTED FACES
BAG
Detected labels:
- Elijah Wood
- Karl Urban
- Andy Serkis

Figure 1. *Labeled Yahoo! News* document with the automatically detected faces and labels on the right. The bag contains 4 instances and 3 labels; the name of Liv Tyler was not detected from text.

Some authors [11, 12] have proposed to learn a distance metric in such weakly supervised contexts where the labels (*e.g.*, tags) are provided only at the image level. Inspired by a multiple instance learning (MIL) formulation [6] where the objects to be compared are sets (called bags) that contain one or multiple instances, they learn a metric so that the distances between similar bags (*i.e.*, bags that contain instances in the same category) are smaller than the distances between dissimilar bags (*i.e.*, none of their instances are in the same category). In the context of Fig. 1, the instances of a bag are the feature vectors of the faces extracted in the image with a face detector [28]. Two bags are considered similar if at least one person is labeled to be present in both images; they are dissimilar otherwise. In the context of image annotation [12] (*e.g.*, in the Corel5K dataset), a bag is an image and its instances are image regions extracted with an image segmentation algorithm [25]. The similarity between bags also depends on the co-occurrence of at least one tag provided at the image level.

Multiple Instance Metric Learning (MIML) approaches [11, 12] decompose the problem into two steps: (1) they first determine and select similar instances in the different training bags, (2) and then solve a classic metric learning problem over the selected instances. The optimization of these two steps is done alternately, which is suboptimal, and the metric learning approaches that they use in the second step have high complexity and may thus not be scalable.

Contributions: In this paper, we propose a MIML method that jointly learns a metric and the assignment of instances in a MIL context by exploiting weakly supervised labels. In particular, our approach jointly learns the two steps of MIML approaches [11, 12] by formulating the set of instances as a function of the learned metric. We also present a nonlinear kernel extension of the model. Our method obtains state-of-the-art performance for the standard tasks of weakly supervised face recognition and automated image annotation. It also has better algorithmic complexity than classic MIML approaches and is much faster.

2. Proposed Model

In this section, we present our approach that we call *Multiple Instance Metric Learning for Cluster Analysis* (MIMLCA) which learns a metric in weakly supervised multi-label contexts. We first introduce our notation and variables. We explain in Section 2.2 how our model infers which instances in the dataset are similar when both the sets of labels in the respective bags and the distance metric to compare instances are known and fixed. Finally, we present our distance metric learning algorithm in Section 2.3.

2.1. Preliminaries and notation

Notation: $\mathbb{S}_+^d$ is the set of $d \times d$ symmetric positive semidefinite (PSD) matrices. We note $\langle A, B \rangle := \text{tr}(AB^\top)$, the Frobenius inner product where A and B are real-valued matrices; and $\|A\| := \sqrt{\text{tr}(AA^\top)}$, the Frobenius norm of A. $\mathbf{1}$ is the vector of all ones with appropriate dimensionality and $A^\dagger$ is the Moore-Penrose pseudoinverse of A.

Model: As in most distance metric learning work [14], we consider the Mahalanobis distance metric d_M that is parameterized by a $d \times d$ symmetric PSD matrix $M = LL^\top$ and is defined for all $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ as:

$$\mathsf{d}_M(\mathbf{a}, \mathbf{b}) = \sqrt{(\mathbf{a} - \mathbf{b})^\top M (\mathbf{a} - \mathbf{b})} = \|(\mathbf{a} - \mathbf{b})^\top L\| \quad (1)$$

Training data: We consider the setting where the training dataset is provided as m (weakly) labeled bags. In detail, each bag $X_i \in \mathbb{R}^{n_i \times d}$ contains n_i instances, each of which is represented as a d-dimensional feature vector. The whole training dataset can thus be assembled into a single matrix $X = [X_1^\top, \cdots, X_m^\top]^\top \in \mathbb{R}^{n \times d}$ that concatenates the m bags and where $n = \sum_{i=1}^m n_i$ is the total number of instances. We assume that (a subset of) the instances in X belong to (a subset of) k training categories. In the weakly supervised MIL setting that we consider, we are provided with the bag label matrix $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_m]^\top \in \{0,1\}^{m \times k}$, where Y_{ic} (*i.e.*, the c-th element of $\mathbf{y}_i \in \{0,1\}^k$) is 1 if the c-th category is a candidate category for the i-th bag (*i.e.*, the c-th category is labeled as being present in the i-th bag), and 0 otherwise. For instance, the matrix Y is extracted from the image tags in the image annotation task, and extracted from text in the *Labeled Yahoo! News* dataset (see Fig. 1).

Instance assignment: As the annotations in Y are provided at the image level (*i.e.*, we do not know exactly the labels of the instances in the bags), our method has to perform inference to determine the categories of the instances in X. We then introduce the instance assignment matrix $H \in \{0,1\}^{n \times k}$ which is *not* observed and that we want to infer. In the following, we write our inference problem so that $H_{jc} = 1$ if the j-th instance is inferred to be in category c, and 0 otherwise. We also assume that although a bag can contain multiple categories, each instance is supposed to belong to *none or one* of the k categories.

In many settings, as labels may be extracted automatically, some categories may be mistakenly labeled as present in some bags, or they may be missing (see Fig. 1). Many instances also belong to none of the k training categories and should thus be left unassigned. Following [11] and [12], if a bag is labeled as containing a specific category, we assign at most one instance of the bag to the category; this makes the model robust to the possible noise in annotations. In the ideal case, all the candidate categories and training instances can be assigned and we then have $\forall i, \mathbf{y}_i^\top \mathbf{1} = n_i$. However, in practice, due to uncertainty or detection errors, it could happen that $\mathbf{y}_i^\top \mathbf{1} < n_i$ (*i.e.*, some instances in the i-th bag are left unassigned) or $\mathbf{y}_i^\top \mathbf{1} > n_i$ (*i.e.*, some labels in the i-th bag do not correspond to any instance).

Reference vectors: We also consider that each category $c \in \{1, \cdots, k\}$ has a representative vector $\mathbf{z}_c \in \mathbb{R}^d$ that we call reference vector. Our goal is to learn both M and the reference vectors so that all the instances inferred to be in a category are closer to the reference vector of their respective category than to any other reference vector (whether they are representatives of candidate categories or not). In the following, we concatenate all the reference vectors into a single matrix $Z = [\mathbf{z}_1, \ldots, \mathbf{z}_k]^\top \in \mathbb{R}^{k \times d}$. We show in Section 2.2 that the optimal value of Z can be written as a function of X, H and M.

Before introducing our metric learning approach, we explain how inference is performed when d_M is fixed.

2.2. Weakly Supervised Multi-instance kmeans

We now explain how our method based on kmeans performs inference on a given set of bags X in our weakly supervised setting. The goal is to assign the instances in X to candidate categories by exploiting both the provided bag label matrix Y and a (fixed) Mahalanobis distance metric d_M. We show in Eq. (7) that our kmeans problem can be reformulated as predicting a single clustering matrix.

To assign the instances in X to the candidate categories (whose presence in the respective bags is known thanks to Y), one natural method is to assign each instance in X to its closest reference vector $\mathbf{z}_c$ belonging to a candidate category. Given the bags X and the provided bag label matrix $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_m]^\top \in \{0,1\}^{m \times k}$, the goal of our method is then to infer both the instance assignment matrix H and reference vector matrix Z that satisfy the conditions mentioned in Section 2.1. Therefore, we constrain H to belong to the following consistency set:

$$\mathcal{Q}^{\mathcal{V}} := \{H = [H_1^\top, \cdots, H_m^\top]^\top : \forall i,\ H_i \in \mathcal{V}_i\} \qquad (2)$$

$$\mathcal{V}_i := \{H_i \in \{0,1\}^{n_i \times k} : H_i\mathbf{1} \leq \mathbf{1}, H_i^\top \mathbf{1} \leq \mathbf{y}_i, \mathbf{1}^\top H_i \mathbf{1} = p_i\}$$

where H_i is the assignment matrix of the n_i instances in the i-th bag, and $p_i := \min\{n_i, \mathbf{y}_i^\top \mathbf{1}\}$. The first condition $H_i\mathbf{1} \leq \mathbf{1}$ implies that each instance is assigned to *at most* one category. The second condition $H_i^\top \mathbf{1} \leq \mathbf{y}_i$, together with the last condition $\mathbf{1}^\top H_i \mathbf{1} = p_i$, ensures that at most one instance in a bag is assigned to each candidate category (*i.e.*, the categories c satisfying $Y_{ic} = 1$).

For a fixed metric d_M, our method finds the assignment matrix $H \in \mathcal{Q}^{\mathcal{V}}$ for the training bags $X \in \mathbb{R}^{n \times d}$ and the vectors $Z = [\mathbf{z}_1, \ldots, \mathbf{z}_k]^\top \in \mathbb{R}^{k \times d}$ that minimize:

$$\min_{H \in \mathcal{Q}^{\mathcal{V}}, Z \in \mathbb{R}^{k \times d}} \sum_{j=1}^{n} \sum_{c=1}^{k} H_{jc} \cdot \mathsf{d}_M^2(\mathbf{x}_j, \mathbf{z}_c) \qquad (3)$$

$$= \min_{H \in \mathcal{Q}^{\mathcal{V}}, Z \in \mathbb{R}^{k \times d}} \| \operatorname{diag}(H\mathbf{1})XL - HZL\|^2 \qquad (4)$$

where $\mathbf{x}_j$ is the j-th instance (*i.e.*, $\mathbf{x}_j^\top$ is the j-th row of X) and d_M is the Mahalanobis distance defined in Eq. (1) with $M = LL^\top$. The goal of Eq. (3) is to assign the instances in X to the closest reference vectors of candidate categories while satisfying the constraints defined in Eq. (2).

The details of the current paragraph can be found in the supp. material, Section A.1. Our goal is to rewrite problem (3) in a convenient way as a function of one variable. As Z is unconstrained in Eq. (4), its minimizer can be found in closed-form: $Z = H^\dagger XLL^\dagger$ [34, Example 2]. From its formulation, we observe that $ZL = H^\dagger XL$ is the set of k mean vectors (*i.e.*, centroids) of the instances in X assigned to the k respective clusters and mapped by L. By plugging the closed-form expression of Z into Eq. (4), the kmeans method in Eq. (4) is equivalent to the following problems:

$$\min_{H \in \mathcal{Q}^{\mathcal{V}}} \| \operatorname{diag}(H\mathbf{1})XL - HH^\dagger XL\|^2 \qquad (5)$$

$$\Leftrightarrow \max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XMX^\top \rangle, \qquad (6)$$

where we define $\mathcal{P}^{\mathcal{V}}$ as $\mathcal{P}^{\mathcal{V}} := \{I + HH^\dagger - \operatorname{diag}(H\mathbf{1}) : H \in \mathcal{Q}^{\mathcal{V}}\}$ and I is the identity matrix. Note that all the matrices in $\mathcal{P}^{\mathcal{V}}$ are orthogonal projection matrices (hence symmetric PSD). For a fixed Mahalanobis distance matrix M, we have reduced the weakly supervised multi-instance kmeans formulation (3) into optimizing a linear function over the set $\mathcal{P}^{\mathcal{V}}$ in Eq. (6). We then define the following prediction rule applied on the set of training bags X:

$$f_{M,\mathcal{P}^{\mathcal{V}}}(X) := \operatorname*{arg\,max}_{A \in \mathcal{P}^{\mathcal{V}}} \ \langle A, XMX^\top \rangle \qquad (7)$$

which is the set of solutions of Eq. (6). We remark that our prediction rule in Eq. (7) assumes that the candidate categories for each bag are known (via $\mathcal{V}_i$).

2.3. Multi-instance Metric Learning for Clustering

We now present how to learn M so that the clustering obtained with d_M is as robust as possible to the case where the candidate categories are unknown. We first write our problem as learning a distance metric so that the clustering predicted when knowing the candidate categories (*i.e.*, Eq. (7)) is as similar as possible to the clustering predicted when the candidate categories are unknown. We then relax our problem and show that it can be solved efficiently.

Our goal is to learn M so that the closest reference vector (among the k categories) of any assigned instance is the reference vector of one of its candidate categories. In this way, an instance can be assigned even when its candidate categories are unknown, by finding its closest reference vector w.r.t. d_M. A good metric d_M should then produce a sensible clustering (*i.e.*, solution of Eq. (7)) even when the set of candidate categories is unknown. To achieve this goal, we consider the set of predicted assignment matrices $Q^{\mathcal{G}}$ (instead of $Q^{\mathcal{V}}$) which ignores Y and where $\mathcal{G}$ is defined as:

$$\mathcal{G}_i := \{H_i \in \{0,1\}^{n_i \times k} : H_i\mathbf{1} \leq \mathbf{1}, \mathbf{1}^\top H_i \mathbf{1} = p_i\} \qquad (8)$$

With $Q^{\mathcal{G}}$, the $\tilde{n} = \mathbf{1}^\top H\mathbf{1}$ assigned instances can be assigned to any of the k training categories instead of only the candidate categories. We want to learn $M \in \mathbb{S}_+^d$ so that the clustering $f_{M,\mathcal{P}^{\mathcal{G}}}$ obtained under the non-informative signal $\mathcal{G}$ is as similar as possible to the clustering $f_{M,\mathcal{P}^{\mathcal{V}}}$ under the weak supervision signal $\mathcal{V}$. Our approach then aims at finding $M \in \mathbb{S}_+^d$ that maximizes the following problem:

$$\max_{M \in \mathbb{S}_+^d} \min_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \min_{\hat{C} \in f_{M,\mathcal{P}^{\mathcal{G}}}(X)} \langle C, \hat{C} \rangle \qquad (9)$$

where C and $\hat{C}$ are clusterings obtained with d_M using different weak supervision signals $\mathcal{V}$ and $\mathcal{G}$. We note that the similarity $\langle C, \hat{C}\rangle$ is in $[0, n]$ as C and $\hat{C}$ are both $n \times n$ orthogonal projection matrices. In the ideal case, Eq. (9) is maximized when the optimal $\hat{C}$ equals the optimal C. In this case, the closest reference vectors of assigned instances are reference vectors of candidate categories. Eq. (9) can actually be seen as a large margin problem as explained in the supp. material, Section A.2.

Since optimizing over $\mathcal{P}^{\mathcal{G}}$ is difficult, we simplify the problem by using spectral relaxation [22, 32, 35]. Instead of constraining $\hat{C}$ to be in $f_{M,\mathcal{P}^{\mathcal{G}}}(X)$, we replace $\mathcal{P}^{\mathcal{G}}$ with its superset $\mathcal{N}$ defined as the set of $n \times n$ orthogonal projection matrices. In other words, we constrain $\hat{C}$ to be in $f_{M,\mathcal{N}}(X)$. The set $f_{M,\mathcal{N}}(X) := \arg\max_{A \in \mathcal{N}} \langle A, XMX^\top\rangle$ is the set of orthogonal projectors onto the leading eigenvectors of $XMX^\top$ [9, 21]. However, just as in PCA, not all the eigenvectors need to be kept. We then propose to select the eigenvectors that lie in the linear space spanned by the columns of the matrix $XMX^\top$ (*i.e.*, in its column space), and ignore eigenvectors in its left null space. For this purpose, we constrain $\hat{C}$ to be in the following relaxed set: $g_M(X) = \{B : B \in f_{M,\mathcal{N}}(X), \operatorname{rank}(B) \leq \operatorname{rank}(XMX^\top)\}$. Our relaxed version of problem (9) is then written:

$$\max_{M \in \mathbb{S}^d_+} \min_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C}\rangle \tag{10}$$

Theorem 2.1. *A global optimum matrix $C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)$ in problem (10) is found by solving the following problem:*

$$C \in \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XX^\dagger\rangle \tag{11}$$

The proof can be found in the supp. material, Section A.3. Finding C in Eq. (11) corresponds to solving an adaptation of kmeans (see supp. material, Section A.4):

$$\min_{H \in \mathcal{Q}^{\mathcal{V}}, Z=[\mathbf{z}_1,\cdots,\mathbf{z}_k]^\top \in \mathbb{R}^{k\times s}} \sum_{j=1}^{n}\sum_{c=1}^{k} H_{jc} \cdot \|\mathbf{u}_j - \mathbf{z}_c\|^2, \tag{12}$$

where $\mathbf{u}_j^\top$ is the j-th row of $U \in \mathbb{R}^{n\times s}$ which is a matrix with orthonormal columns such that $s := \operatorname{rank}(X)$ and $XX^\dagger = UU^\top$. To solve Eq. (12), we use an adaptation of Lloyd's algorithm [19] illustrated in Algorithm 1 where $U_i \in \mathbb{R}^{n_i \times s}$ is a submatrix of U and represents the eigenrepresentation of the bag $X_i \in \mathbb{R}^{n_i \times d}$. As explained in the supp. material, Algorithm 1 minimizes Eq. (12) by alternately optimizing over Z and H. Convergence guarantees of Algorithm 1 are studied in the supp. material.

Once an optimal instance assignment matrix $H \in \mathcal{Q}^{\mathcal{V}}$ has been inferred, we can use any type of classifier or metric learning approach to discriminate the different categories. We propose to use the approach in [18] which learns a metric d_M in the context where each object is a bag that contains one instance and there is only one candidate category for each bag. It can be viewed as a special case of Eq. (10)

Algorithm 1 MIML for Cluster Analysis (MIMLCA)

```
input : Training set $X \in \mathbb{R}^{n\times d}$, training labels $Y \in \{0,1\}^{m\times k}$
 1: Create $U = [U_1^\top, \cdots, U_m^\top]^\top \in \mathbb{R}^{n\times s}$ s.t. $s = \operatorname{rank}(X)$, $XX^\dagger = UU^\top, \forall i \in \{1,\cdots,m\}\ U_i \in \mathbb{R}^{n_i\times s}$
 2: Initialize assignments (e.g., randomly): $H \in \mathcal{Q}^{\mathcal{V}}$
 3: repeat
 4:     let $\mathbf{h}_c$ be the $c$-th column of $H$, $\frac{\mathbf{h}_c^\top}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}$ is the $c$-th row of $H^\dagger$
 5:     $Z \leftarrow H^\dagger U \in \mathbb{R}^{k\times s}$
 6:     For each bag $i = 1$ to $m$, $H_i \leftarrow \mathsf{assign}(U_i, Z, Y)$ % solve Eq. (13)
 7:     $H \leftarrow [H_1^\top, \cdots, H_m^\top]^\top \in \mathcal{Q}^{\mathcal{V}}$
 8: until convergence
 9: % Select the rows $j$ of $X$ and $H$ for which $\sum_c H_{jc} = 1$. We use the logical indexing Matlab notation: $H\mathbf{1}$ is a Boolean vector/logical array. $A(H\mathbf{1},:)$ is the submatrix of $A$ obtained by dropping the zero rows of $H$ (i.e. dropping the rows of $A$ corresponding to the indices of the false elements of $H\mathbf{1}$) while keeping all the columns of $A$.
10: $X \leftarrow X(H\mathbf{1},:)$, $n \leftarrow \mathbf{1}^\top H\mathbf{1}$, $H \leftarrow H(H\mathbf{1},:)$
11: $M \leftarrow X^\dagger HH^\dagger(X^\dagger)^\top$
```

where $\{C\} = f_{M,\mathcal{P}^{\mathcal{V}}}(X)$ is a singleton that does not depend on M (*i.e.*, the same matrix C is returned for any value of M) and $\hat{C}$ is now constrained to be in the set: $\{B : B \in f_{M,\mathcal{N}}(X), \operatorname{rank}(B) = \operatorname{rank}(C), C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)\}$ as the rank of C (and thus of $\hat{C}$) is now known. An optimal Mahalanobis matrix in this case is $M = X^\dagger C(X^\dagger)^\top$ [18].

In detail, Algorithm 1 first creates in step 1 the matrix U whose columns are the left-singular vectors of the nonzero singular values of X. Next, Algorithm 1 alternates between computing the centroids Z (step 5) and inferring the instance assignment matrix H (steps 6-7). The latter step is decoupled among the m bags; the function $\mathsf{assign}(U_i, Z, Y)$ returns a solution of the following assignment problem:

$$H_i \in \arg\min_{G \in \mathcal{V}_i} \|\operatorname{diag}(G\mathbf{1})U_i - GZ\|^2, \tag{13}$$

which is solved exactly with the Hungarian algorithm [13] by exploiting the cost matrix that contains the squared Euclidean distances between the rows of U_i and the centroids $\mathbf{z}_c$ for which $Y_{ic} = 1$. Let us note $q_i := \max\{n_i, \mathbf{y}_i^\top\mathbf{1}\}$, computing the cost matrix costs $O(sp_iq_i)$ and the Hungarian algorithm costs in practice $O\left(p_i^2 q_i\right)$ [3]. It is efficient in our experiments as q_i is small ($\forall i, p_i \leq q_i \leq 15$).

In conclusion, we have proposed an efficient metric learning algorithm that takes weak supervision into account. We explain below how to extend it to the nonlinear case.

Nonlinear Kernel Extension: We now briefly explain how to learn a nonlinear Mahalanobis metric by using kernels [24]. We first consider the case where each bag contains a single instance and has only one candidate category, this case corresponds to [18] (*i.e.*, steps 10-11 of Algo 1).

Let k be a kernel function whose feature map $\phi(\cdot)$ maps the instance $\mathbf{x}_j$ to $\phi(\mathbf{x}_j)$ in some reproducing kernel Hilbert space (RKHS) $\mathcal{H}$. Using the generalized representer theorem [23], we can write the Mahalanobis matrix M (in the RKHS) as: $M = \Phi P^\top P \Phi^\top$, where $\Phi =$

$[\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_n)]$ and $P \in \mathbb{R}^{k \times n}$. Let $K \in \mathbb{S}_+^n$ be the kernel matrix on the training instances: $K = \Phi^\top \Phi$, where $K_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \mathsf{k}(\mathbf{x}_i, \mathbf{x}_j)$. Eq. (7) is then written:

$$f_{(\Phi P^\top P \Phi^\top), \mathcal{P}^\mathcal{V}}(\Phi^\top) = \underset{A \in \mathcal{P}^\mathcal{V}}{\arg\max} \langle A, K P^\top P K \rangle \tag{14}$$

A solution of [18, Eq. (13)] is $M = \Phi K^\dagger J (\Phi K^\dagger J)^\top$ where $JJ^\top = HH^\dagger$ is the desired clustering matrix.[1] We then replace Step 11 of Algo 1 by $M \leftarrow \Phi K^\dagger J (\Phi K^\dagger J)^\top$.

To extend Eq. (11) to the nonlinear case in the MIL context, the matrix $U \in \mathbb{R}^{n \times s}$ in step 1 can be formulated as $UU^\top = KK^\dagger$ where $s = \text{rank}(K)$. Note that $XX^\dagger = XX^\top (XX^\top)^\dagger = KK^\dagger$ when $\forall \mathbf{x},\ \phi(\mathbf{x}) = \mathbf{x}$.

The complexity of our method is $O(nd \min\{d, n\})$ in practice: it is linear in the number of instances n and quadratic in the dimensionality d as $d < n$ in our experiments (see details in supp. material, Section A.5).

3. Related work

MIL was introduced in the context of drug activity prediction [6] to distinguish positive bags from negative bags. Most MIL problems [1, 4, 5, 10, 27, 36, 37] consider only 2 categories: bags are considered either positive or negative. In this paper, we focus on multi-label contexts (*i.e.*, $k \geq 2$) wherein MIML approaches were proven successful.

MIML: the Mahalanobis distance was already used [11, 12] in the weakly supervised context where the objects to be compared are bags containing multiple instances and the category membership labels of instances are provided at bag-level. Jin *et al.* [12] learn a distance metric optimized to group similar instances from different bags into common clusters. Their method decomposes their learning algorithm into three sets of variables which are: (1) the reference vectors (called centroids) of their categories, (2) an assignment matrix that determines instances that are closest to the centroids of their categories, (3) their Mahalanobis distance metric d_M. They use an iterative algorithm that alternates the optimization over these three sets of variables and has high algorithmic complexity. Our approach also decomposes the problem into three variables, but our variables can all be written as a function of each other, which means that we only have to optimize the problem over one variable to get the formulation of the other variables. In this way, all the variables of our method are learned jointly, and optimizing over them has low computational complexity (*i.e.*, the complexity of our method is $O(nd^2)$). Moreover, the method in [12] is not appropriate for nonlinear kernelized Mahalanobis distances as it explicitly formulates centroids and optimizes over them; this is problematic if the codomain of the (kernel) feature map is infinite-dimensional (*e.g.*, most RBF kernels) or even high-dimensional.

Guillaumin *et al.* [11] also consider weak supervision: their metric is learned so that distances between the closest instances of similar bags are smaller than distances between instances of dissimilar bags. As in [12], their method suffers from the decomposition of the similarity matching of instances and the learned metric as they depend on each other. Moreover, they only consider local matching between pairs of bags instead of global matching of the whole dataset to group similar instances into common clusters. Furthermore, as mentioned in [11, Section 5] and unlike our approach, their method does not scale linearly in n.

Wang *et al.* [29] learn multiple metrics (one per category) in a MIL setting. For each category, their distance is the average distance between all the instances in bags that contain the category and their respective closest instance in a given bag. As all the instances in bags that contain a given category are taken into account, their Class-to-Bag (C2B) method is less robust to outlier instances than our method that assigns at most one instance per bag to a candidate category. Their method is then not appropriate for contexts such as face recognition where a small proportion of instances in the different bags is relevant to the category. Moreover, their method requires subsampling a large number of constraints to be scalable. Indeed, their complexity is linear in the number of instances n thanks to subsampling and the complexity of each iteration of their iterative algorithm is cubic in the dimensionality d.

Closed-form training in the supervised setting: In the fully supervised context where each object can be seen as a bag that contains only one instance and where the label of each instance is provided without uncertainty, an efficient metric learning approach optimized to group a set of vectors into k desired clusters was proposed in [18]. The method assumes that the ground truth partition of the training set is known. It finds an optimal metric such that the partition obtained by applying kmeans with the metric is as close as possible to the ground truth partition. In contrast, our approach extends [18] to the weakly supervised case where the objects are multiple instance bags and the ground truth clustering assignment is unknown. A main difficulty is that the set of candidate assignment matrices $\mathcal{Q}^\mathcal{V}$ in Eq. (2) that satisfy the provided weak annotations can be very large. Moreover, [18] did not provide a criterion to determine which matrix in $\mathcal{Q}^\mathcal{V}$ is optimal in our context.

Our contribution wrt [18] includes: 1) the kmeans adaptation to optimize over weakly supervised bags (Section 2.2), 2) the derivation of the (relaxed) metric learning problem to learn a metric that is robust to the case where the bag labels are not provided, 3) the efficient algorithm (Algorithm 1) that returns the optimal assignment matrix, 4) a nonlinear kernel version.

[1] A matrix J such that $JJ^\top = HH^\dagger$ and $H \in \mathcal{Q}^\mathcal{V}$ can be computed efficiently: let $\mathbf{h}_c$ be the c-th column of H, then the c-th column of J can be written $\mathbf{j}_c = \frac{1}{\sqrt{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}}} \mathbf{h}_c$.

4. Experiments

We evaluate our method called MIMLCA in the face identification and image annotation tasks where the dataset is labeled in a weakly supervised way. We implemented our method in Matlab and ran the experiments on a 2.6GHz machine with 4 cores and 16GB of RAM.

4.1. Weakly labeled face identification

We use the subset of the *Labeled Yahoo! News* dataset[2] introduced in [2] and manually annotated by [11] for the context of face recognition with weak supervision. The dataset is composed of 20,071 documents containing a total of 31,147 faces detected with a Viola-Jones face detector [28]. The number of categories (*i.e.*, identified persons) is $k = 5,873$ (mostly politicians and athletes). An example document is illustrated in Fig. 1. Each document contains an image and some text, it also contains at least one detected face or name in the text. Each face is represented by a d-dimensional vector where $d = 4,992$. 9,594 of the 31,147 detected faces are unknown persons (*i.e.*, they belong to none of the k training categories), undetected names or not face images. As already explained, we consider documents as bags and detected faces as instances. See supp. material, Section A.7 for additional details on the dataset.

Setup: We randomly partition the dataset into 10 equal sized subsets to perform 10-fold cross-validation: each subset then contains 2,007 documents (except one that contains 2,008 documents). The training dataset of each split thus contains $m \approx 18,064$ documents and $n \approx 28,000$ faces.

Classification protocol: To compare the different methods, we consider two evaluation metrics: the average classification accuracy across all training categories and the *precision* (defined in [11] as the ratio of correctly named faces over the total number of faces in the test dataset). At test time, a face whose category membership is known is assigned to one of the $k = 5,873$ categories. To avoid a strong bias of the evaluation metrics due to under-represented categories, we classify at test time only the instances in categories that contain at least 5 elements in the test dataset (this arbitrary threshold seemed sensible to us as it is small enough without being too small). This corresponds to selecting about 50 test categories (depending on the split). We note that test instances can be assigned to any of the k categories and not only to the 50 selected categories.

Scenarios/Settings: To train the different models, we consider the same three scenarios/settings as [11]:

(a) *Instance-level ground truth.* We know here for each training instance its actual category; it corresponds to a supervised single-instance context. In this setting, our method is equivalent to MLCA [18] and provides an upper bound on the performance of models learned with weak supervision.

(b) *Bag-level ground truth.* The presence of identified persons in an image is provided at bag-level by humans, which corresponds to a weakly supervised context.

(c) *Bag-level automatic annotation.* The presence of identified persons in an image is automatically extracted from text. This setting is unsupervised in the sense that it does not require human input and may be noisy. The label matrix Y is automatically extracted as described in Fig. 1.

Classification of test instances: In the task that we consider, we are given the vector representation of a face and the model has to determine which of the k training categories it belongs to. In the linear case, the category of a test instance $\mathbf{x}_t \in \mathbb{R}^d$ can be naturally determined by solving:

$$\arg\min_{c \in \{1,\cdots,k\}} \mathsf{d}_M^2(\mathbf{x}_t, \mathbf{z}_c) \tag{15}$$

where $\mathbf{z}_c$ is the mean vector of the training instances assigned to category c, and d_M is a learned metric.

In the case of MIMLCA, the learned metric (in step 11) can be written $M = LL^\top$ where $L = X^\dagger J$ and J is constructed as explained in Footnote 1. For any training instance $\mathbf{x}_j$ (inferred to be) in category c, the matrix M is then learned so that the maximum element of the vector $(L^\top \mathbf{x}_j) \in \mathbb{R}^k$ is its c-th element and all the other elements are zeros. We can then also use the prediction function:

$$\arg\max_{c \in \{1,\cdots,k\}} \mathbf{x}_t^\top X^\dagger \mathbf{j}_c - \alpha \|L^\top \mathbf{z}_c\|^2 \tag{16}$$

where $\mathbf{j}_c$ is the c-th column of J, the value of $\mathbf{x}_t^\top X^\dagger \mathbf{j}_c$ is the c-th element of $L^\top \mathbf{x}_t$, and $\alpha \in \mathbb{R}$ is a parameter manually chosen (see experiments below). The term $-\alpha \|L^\top \mathbf{z}_c\|^2$ accounts for the fact that the metric is learned with clusters having different sizes. Note that α is not used during training. See supp. material, Section A.6 for the nonlinear case.

Experimental results: Table 1 reports the average classification accuracy across categories and the precision scores obtained by the different baselines and our method in the linear case. Since we are interested in the weakly supervised settings (b) and (c), we cannot evaluate classic metric learning approaches, such as LMNN [31], that require instance-level annotations (*i.e.*, scenario (a)). We reimplemented [12] as best as we could as the code is not available (see supp. material, Section A.10). The codes of the other baselines are publicly available (except [29] that we also reimplemented, see supp. material, Section A.11).

We do not cross-validate our method as it does not have hyperparameters. For all the other methods, to create the best possible baselines, we report the best scores that we obtained on the test set when tuning the hyperparameters. We tested different MIL baselines [1, 4, 5, 10, 29, 36, 37], most of them are optimized for MIL classification in the bi-class case (*i.e.*, when there are 2 categories of bags which

[2]We use the features available at http://lear.inrialpes.fr/people/guillaumin/data.php

Method	Scenario/Setting (see text)	Accuracy (closest centroid)	Precision (closest centroid)	Training time (in seconds)
Euclidean Distance	None	57.0 ± 2.4	56.7 ± 2.0	No training
Linear MLCA [18]	(a) = *Instance gt*	$\mathbf{66.8 \pm 4.2}$	$\mathbf{77.7 \pm 2.2}$	**59**
MIML (our reimplementation of [12])	(b) = *Bag gt*	56.1 ± 3.3	55.5 ± 2.6	17,728
MildML [11]	(b)	54.9 ± 3.6	54.6 ± 3.3	7,352
Linear MIMLCA (ours)	(b)	$\mathbf{65.3 \pm 3.7}$	$\mathbf{76.6 \pm 2.1}$	**163**
MIML (our reimplementation of [12])	(c) = *Bag auto*	52.6 ± 13.0	52.2 ± 13.8	19,091
MildML [11]	(c)	33.9 ± 3.0	31.2 ± 2.9	7,520
Linear MIMLCA (ours)	(c)	$\mathbf{63.2 \pm 4.7}$	$\mathbf{74.9 \pm 3.0}$	**180**

Table 1. Test classification accuracies and precision scores (mean and standard deviation in %) on *Labeled Yahoo! News*

Method	Scenario	Accuracy	Precision	Training time	Scenario	Accuracy	Precision	Training time
MildML [11]	(b)	52.4 ± 4.7	62.2 ± 2.9	7,352 seconds	(c)	55.7 ± 4.4	66.0 ± 2.1	7,520 seconds

Table 2. Test scores of MildML on *Labeled Yahoo! News* when assigning test instances to the category of their closest training instances

Method	Scenario	Eval. metric	$\alpha = 0$	$\alpha = 0.2$	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 1$	$\alpha = 1.2$	Training time
Linear MLCA	(a)	Accuracy	77.6 ± 3.1	88.0 ± 2.2	88.5 ± 2.1	$\mathbf{89.5 \pm 2.0}$	89.3 ± 1.8	88.9 ± 2.0	59 seconds
		Precision	78.0 ± 2.0	88.8 ± 1.3	89.4 ± 1.4	90.8 ± 1.1	$\mathbf{91.5 \pm 1.0}$	91.4 ± 1.0	
Linear MIMLCA	(b)	Accuracy	74.2 ± 2.7	85.9 ± 2.1	86.5 ± 2.0	$\mathbf{87.7 \pm 1.9}$	87.4 ± 1.8	87.1 ± 2.0	163 seconds
		Precision	74.8 ± 1.8	87.0 ± 1.4	87.7 ± 1.3	89.3 ± 1.0	89.9 ± 1.2	$\mathbf{90.0 \pm 1.3}$	
	(c)	Accuracy	69.9 ± 2.5	81.2 ± 2.6	81.9 ± 2.5	83.6 ± 2.3	$\mathbf{83.9 \pm 2.1}$	83.7 ± 2.0	180 seconds
		Precision	71.7 ± 1.5	83.0 ± 1.4	83.8 ± 1.4	85.6 ± 1.4	86.9 ± 1.5	$\mathbf{87.0 \pm 1.5}$	
$\mathrm{k}_{\chi^2}^{\mathrm{RBF}}$ MLCA	(a)	Accuracy	77.2 ± 3.0	94.4 ± 1.6	$\mathbf{94.5 \pm 1.8}$	92.5 ± 2.0	87.1 ± 2.2	84.5 ± 2.9	50 seconds
		Precision	73.6 ± 1.8	95.3 ± 1.0	$\mathbf{95.5 \pm 1.2}$	94.9 ± 1.1	92.3 ± 1.4	91.0 ± 1.7	
$\mathrm{k}_{\chi^2}^{\mathrm{RBF}}$ MIMLCA	(b)	Accuracy	74.0 ± 2.9	92.6 ± 1.8	$\mathbf{92.8 \pm 1.6}$	91.1 ± 2.0	84.5 ± 2.5	82.0 ± 2.6	154 seconds
		Precision	70.6 ± 1.8	93.6 ± 1.2	$\mathbf{94.0 \pm 1.0}$	93.7 ± 1.1	90.6 ± 1.5	89.4 ± 1.6	
	(c)	Accuracy	67.1 ± 2.9	88.2 ± 1.9	$\mathbf{88.5 \pm 2.1}$	87.2 ± 1.8	81.1 ± 3.3	78.6 ± 3.6	172 seconds
		Precision	63.7 ± 1.8	89.0 ± 1.3	89.7 ± 1.5	$\mathbf{90.0 \pm 1.3}$	87.5 ± 2.2	86.3 ± 2.4	

Table 3. Test classification accuracies and precision scores in % of the linear and nonlinear models for the 10-fold cross-validation evaluation for different values of α in Eq. (16)

are "positive" and "negative"); as proposed in [4], we apply for these baselines the one-against-the-rest heuristic to adapt them to the multi-label context. However, there are more than 5,000 training categories. Since most categories contain very few examples and these baselines learn classifiers independently, the scale of classification scores may differ. They then obtain less than 10% accuracy and precision in this task (see supp. material, Section A.8 for scores).

Table 1 reports the test performance of the different different methods when assigning a test instance to the category with closest centroid w.r.t. the metric (*i.e.*, using the prediction function in Eq. (15)). We use this evaluation because (MI)MLCA and MIML [12] are learned to optimize this criterion. The set of centroids exploited by MIMLCA in settings (b) and (c) is determined in Algorithm 1. MIML also exploits the set of centroids that it learns. To evaluate MildML and the Euclidean distance, we exploit the ground truth instance centroids (*i.e.*, the mean vectors of instances in the k categories in the context where we know the category of each instance) although these ground truth centroids are normally not available in settings (b) and (c) as annotations are provided at bag-level and not at instance-level.

In Table 2, a test instance is assigned to the category of the closest training instance w.r.t. the metric. We use this evaluation as MildML is optimized for this criterion although the category of the closest training instance is normally available only in setting (a). MildML then improves its precision scores compared to Table 1.

We see in Table 1 that our linear method MIMLCA learned in weakly supervised scenarios (b) and (c) performs almost as well as the fully supervised model MLCA [18] in setting (a). Our method can then be learned fully automatically in scenario (c) at the expense of a slight loss in accuracy. Moreover, our method learned with scenario (c) outperforms other MIL baselines learned with scenario (b).

Nonlinear model: Table 3 reports the recognition performances of (MI)MLCA in the linear and nonlinear cases when we exploit the prediction function in Eq. (16) for different values of α. In the nonlinear case, we choose the generalized radial basis function (RBF) $\mathrm{k}_{\chi^2}^{\mathrm{RBF}}(\mathbf{a}, \mathbf{b}) = e^{-D_{\chi^2}^2(\mathbf{a},\mathbf{b})}$ where $\mathbf{a}$ and $\mathbf{b}$ are ℓ_1-normalized and $D_{\chi^2}^2(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{d} \frac{(a_i - b_i)^2}{a_i + b_i}$. This kernel function is known to work well for face recognition [20]. With the RBF kernel, we reach 90% classification accuracy and precision. We observe a gain in accuracy of about 5% with the nonlinear version compared to the linear version when $\alpha \simeq 0.25$.

Training times: Tables 1 to 3 report the wall-clock training time of the different methods. We assume that the matrices X and Y (and K in the nonlinear case) are already loaded in memory. Both MLCA and MIMLCA are efficient as they are trained in less than 5 minutes. MIMLCA is 3 times slower than MLCA because it requires computing 2 (economy size) SVDs to compute U and $X^\dagger$ (steps 1 and 11 of Algo 1), each of them takes about 1 minute, whereas MLCA requires only one SVD. Moreover, besides the two SVDs already mentioned, MIMLCA performs an adapted `kmeans` (steps 3 to 8 of Algo 1) which takes less than 1

Method	OE (↓)	Cov. (↓)	AP (↑)	Training time (↓)	Method	OE (↓)	Cov. (↓)	AP (↑)	Training time (↓)
MIMLCA (ours)	**0.516**	**4.829**	**0.575**	**24 seconds**	MILES [4]	0.722	7.626	0.412	511 seconds
MIML (best scores reported in [12])	0.565	5.507	0.535	Not available	miSVM [1]	0.790	9.730	0.261	504 seconds
MIML (our reimplementation of [12])	0.673	6.403	0.462	884 seconds	MILBoost [36]	0.948	13.412	0.174	106 seconds
MildML [11]	0.619	5.646	0.499	59 seconds	EM-DD [37]	0.892	10.527	0.239	38,724 seconds
Citation-kNN [30] (Euclidean dist.)	0.595	5.559	0.513	No training	MInD [5] (*meanmin*)	0.759	8.246	0.373	103 seconds
M-C2B [29]	0.691	6.968	0.440	211 seconds	MInD [5] (*minmin*)	0.703	7.337	0.424	138 seconds
Minimax MI-Kernel [10]	0.734	7.955	0.398	172 seconds	MInD [5] (*maxmin*)	0.721	7.857	0.413	95 seconds

Table 4. Annotation performance on the Corel5K dataset, ↓: the lower the metric, the better, ↑: the larger the metric, the better. OE: One-error, Cov.: Coverage. AP: Average Precision (see definitions in [12, Section 5.1])

minute: the adapted kmeans converges in less than 10 iterations and each iteration takes 5 seconds. We note that our method is one order of magnitude faster than MildML.

In conclusion, our weakly supervised method outperforms the current state-of-the-art MIML methods both in recognition accuracy and training time. It is worth noting that if we apply mean centering on X then the matrix U, whose columns form an orthonormal basis of X, contains the eigenfaces [26] of the training face images (one eigenface per row). Our approach then assigns instances to clusters depending on their distance in the eigenface space.

4.2. Automated image annotation

We next evaluate our method using the same evaluation protocol as [12] in the context of automated image annotation. We use the dataset[3] of Duygulu *et al.* [7] which includes 4,500 training images and 500 test images selected from the UCI Corel5K dataset. Each image was segmented into no more than 10 regions (*i.e.*, instances) by Normalized Cut [25], and each region is represented by a d-dimensional vector where $d = 36$. The image regions are clustered into 500 blobs using kmeans, and a total of 371 keywords was assigned to 5,000 images. As in [12], we only consider the $k = 20$ most popular keywords since most keywords are used to annotate a small number of images. In the end, the dataset that we consider includes $m = 3,947$ training images containing $n = 37,083$ instances, and 444 test images.

To annotate test images, we evaluate our method in the same way as [12] by including our metric in the citation-kNN [30] algorithm which adapts kNN to the multiple instance problem. The citation-kNN [30] algorithm proposes different extensions of the Hausdorff distance to compute distances between bags that contain multiple instances. As proposed in [30], we tested both the *Maximal* and *Minimal Hausdorff distances* (see definitions in [30, Section 2]). For example, the *Minimal Hausdorff Distance* between two bags E and F is the smallest distance between the instances of the different bags: $D_{\min}(E, F) = D_{\min}(F, E) = \min_{\mathbf{e} \in E} \min_{\mathbf{f} \in F} \mathsf{d}_M(\mathbf{e}, \mathbf{f})$ where $\mathbf{e}$ and $\mathbf{f}$ are instances of the bags E and F, respectively. In [30], d_M is the Euclidean distance, we replace it by the different learned metrics of MIML approaches in the same way as [12].

Given a test bag E, we define its references as the r nearest bags in the training set, and its citers as the training bags for which E is one of the c nearest neighbors. The class label of E is decided by a majority vote of the r reference bags and c citing bags. We follow the exact same protocol as [12] and use the same evaluation metrics (see definitions in [12, Section 5.1]). We report in Table 4 the results obtained with minimal Hausdorff distances since they obtained the best performances for all the metric learning methods. As in [12], we tested different values of $c = r \in \{5, 10, 15, 20\}$ and report the results for $c = r = 20$ as they performed the best for all the methods.

We tuned all the baselines and report their best scores on the test set. Our method outperforms the other MIL approaches w.r.t. all the evaluation metrics and it is faster. Our method can then also be used for image annotation.

5. Conclusion

We have presented an efficient MIML approach optimized to perform clustering. Unlike classic MIL approaches, our method does not alternate the optimization over the learned metric and the assignment of instances. Our method only performs an adaptation of kmeans over the rows of the matrix U whose columns form an orthonormal basis of X. Our method is much faster than classic approaches and obtains state-of-the-art performance in the face identification (in the weakly supervised and fully unsupervised cases) and automated image annotation tasks.

Acknowledgments: We thank Xiaodan Liang and the anonymous reviewers for their helpful comments. This work was supported by Samsung and the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DoI/IBC) contract number D16PC00003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/IBC, or the U.S. Government.

[3]We use the features available at http://kobus.ca/research/data/eccv_2002/

References

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, pages 561–568, 2002.

[2] T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y.-W. Teh, E. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *CVPR*, volume 2, pages II–848. IEEE, 2004.

[3] F. Bourgeois and J.-C. Lassalle. An extension of the munkres algorithm for the assignment problem to rectangular matrices. *Communications of the ACM*, 14(12):802–804, 1971.

[4] Y. Chen, J. Bi, and J. Z. Wang. Miles: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1931–1947, 2006.

[5] V. Cheplygina, D. M. Tax, and M. Loog. Multiple instance learning with bag dissimilarities. *Pattern Recognition*, 48(1):264–275, 2015.

[6] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1):31–71, 1997.

[7] P. Duygulu, K. Barnard, J. F. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, pages 97–112. Springer, 2002.

[8] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[9] K. Fan. On a theorem of weyl concerning eigenvalues of linear transformations i. *Proceedings of the National Academy of Sciences of the United States of America*, 35(11):652, 1949.

[10] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *ICML*, volume 2, pages 179–186, 2002.

[11] M. Guillaumin, J. Verbeek, and C. Schmid. Multiple instance metric learning from automatically labeled bags of faces. In *ECCV*, pages 634–647. Springer, 2010.

[12] R. Jin, S. Wang, and Z.-H. Zhou. Learning a distance metric from multi-instance multi-label data. In *CVPR*, pages 896–902. IEEE, 2009.

[13] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[14] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.

[15] R. Lajugie, S. Arlot, and F. Bach. Large-margin metric learning for constrained partitioning problems. In *ICML*, pages 297–305, 2014.

[16] M. T. Law, N. Thome, and M. Cord. Fantope regularization in metric learning. In *CVPR*, pages 1051–1058, June 2014.

[17] M. T. Law, N. Thome, and M. Cord. Learning a distance metric from relative comparisons between quadruplets of images. *IJCV*, 121(1):65–94, 2017.

[18] M. T. Law, Y. Yu, M. Cord, and E. P. Xing. Closed-form training of mahalanobis distance for supervised clustering. In *CVPR*. IEEE, 2016.

[19] S. P. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982, first published in 1957 in a Technical Note of Bell Laboratories.

[20] A. Mignon and F. Jurie. Pcca: A new approach for distance learning from sparse pairwise constraints. In *CVPR*, 2012.

[21] M. L. Overton and R. S. Womersley. Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Mathematical Programming*, 62(1-3):321–357, 1993.

[22] J. Peng and Y. Wei. Approximating k-means-type clustering via semidefinite programming. *SIAM Journal on Optimization*, 18:186–205, 2007.

[23] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Computational learning theory*, pages 416–426. Springer, 2001.

[24] B. Scholköpf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[25] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[26] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Josa a*, 4(3):519–524, 1987.

[27] R. Venkatesan, P. Chandakkar, and B. Li. Simpler non-parametric methods provide as good or better results to multiple-instance learning. In *ICCV*, pages 2605–2613, 2015.

[28] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 4, 2001.

[29] H. Wang, F. Nie, and H. Huang. Robust and discriminative distance for multi-instance learning. In *CVPR*, pages 2919–2924. IEEE, 2012.

[30] J. Wang and J.-D. Zucker. Solving the multiple-instance problem: A lazy learning approach. In *ICML*, pages 1119–1126. Morgan Kaufmann Publishers Inc., 2000.

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

[32] E. P. Xing and M. I. Jordan. On semidefinite relaxations for normalized k-cut and connections to spectral clustering. *Tech Report CSD-03-1265, UC Berkeley*, 2003.

[33] E. P. Xing, M. I. Jordan, S. Russell, and A. Y. Ng. Distance metric learning with application to clustering with side-information. In *NIPS*, pages 505–512, 2002.

[34] Y.-L. Yu and D. Schuurmans. Rank/norm regularization with closed-form solutions: Application to subspace clustering. In *Uncertainty in Artificial Intelligence (UAI)*, 2011.

[35] H. Zha, X. He, C. Ding, M. Gu, and H. D. Simon. Spectral relaxation for k-means clustering. In *NIPS*, pages 1057–1064, 2001.

[36] C. Zhang, J. C. Platt, and P. A. Viola. Multiple instance boosting for object detection. In *NIPS*, pages 1417–1424, 2005.

[37] Q. Zhang and S. A. Goldman. Em-dd: An improved multiple-instance learning technique. In *NIPS*, pages 1073–1080, 2001.

A. Supplementary Material of “Efficient Multiple Instance Metric Learning using Weakly Supervised Data”

A.1. About the reference vectors

A.1.1 Closed-form solution of the reference vectors Z

As mentioned in [34, Example 2], the problem:

$$\min_C \|A - BCD\|^2 \tag{17}$$

can be solved in closed-form: $C = B^\dagger A D^\dagger$.

In Eq. (4), we can write $A = \operatorname{diag}(H\mathbf{1})XL$, $B = H$ and $D = L$. The matrix $Z = H^\dagger \operatorname{diag}(H\mathbf{1})XLL^\dagger$ is then optimal for Eq. (4).

We recall that $H \in \mathcal{Q}^{\mathcal{V}}$. We prove in the following that: $\forall H \in \mathcal{Q}^{\mathcal{V}}, H^\dagger \operatorname{diag}(H\mathbf{1}) = H^\dagger$.

Proof. For any $H \in \mathcal{Q}^{\mathcal{V}}$ satisfying $H\mathbf{1} \neq \mathbf{1}$, there exists a permutation matrix P_π such that $P_\pi H = \begin{bmatrix} \tilde{H} \\ \mathbf{0} \end{bmatrix}$ and $\operatorname{diag}(P_\pi H\mathbf{1}) = \operatorname{diag}\left(\begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}\right)$. Therefore,

$$\begin{aligned} H^\dagger \operatorname{diag}(H\mathbf{1}) &= \left(P_\pi^\top \begin{bmatrix} \tilde{H} \\ \mathbf{0} \end{bmatrix}\right)^\dagger \operatorname{diag}(H\mathbf{1}) = \left(\begin{bmatrix} \tilde{H} \\ \mathbf{0} \end{bmatrix}\right)^\dagger P_\pi \operatorname{diag}(H\mathbf{1}) = \begin{bmatrix} \tilde{H}^\dagger & \mathbf{0} \end{bmatrix} \operatorname{diag}(P_\pi H\mathbf{1}) P_\pi \\ &= \begin{bmatrix} \tilde{H}^\dagger & \mathbf{0} \end{bmatrix} \operatorname{diag}\left(\begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}\right) P_\pi = \begin{bmatrix} \tilde{H}^\dagger & \mathbf{0} \end{bmatrix} P_\pi = H^\dagger. \end{aligned}$$

On the other hand, if $H\mathbf{1} = \mathbf{1}$, $\operatorname{diag}(H\mathbf{1})$ is the identity matrix and we then also have $H^\dagger \operatorname{diag}(H\mathbf{1}) = H^\dagger$. □

It is then clear that $\forall H \in \mathcal{Q}^{\mathcal{V}},\ Z = H^\dagger \operatorname{diag}(H\mathbf{1})XLL^\dagger = H^\dagger XLL^\dagger$ is optimal for Eq. (4).

A.1.2 Mean vector of assigned instances

We explain why $ZL = H^\dagger XLL^\dagger L = H^\dagger XL$ is the set of k mean vectors (*i.e.*, centroids) of the instances in X assigned to the k respective clusters and mapped by L.

By definition, XL is the set of instances in X mapped by L. We note $\mathbf{h}_c$ the c-th column of $H \in \mathcal{Q}^{\mathcal{V}}, \forall c \in \{1, \cdots, k\}$, we can write the c-th row of $H^\dagger = (H^\top H)^\dagger H^\top$ as $\frac{1}{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}} \mathbf{h}_c^\top$ where $\mathbf{h}_c^\top \mathbf{1} = \|\mathbf{h}_c\|^2$ is the number of instances assigned to cluster c. The c-th row of ZL which corresponds to $\mathbf{z}_c^\top L$ can then be written $\mathbf{z}_c^\top L = \frac{1}{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}} \mathbf{h}_c^\top XL$. As $\mathbf{h}_c \in \{0, 1\}^n$, $\mathbf{h}_c^\top XL$ selects and sums the instances assigned to the c-th cluster and mapped by L, $\mathbf{z}_c^\top L = \frac{1}{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}} \mathbf{h}_c^\top XL$ then computes their mean vector (*i.e.*, centroid).

Note that if for some c, $\mathbf{h}_c = \mathbf{0}$, then $(\mathbf{z}_c^\top L)^\top = \mathbf{0}$ is the closest centroid (of a candidate category) to none of the assigned instances as it would otherwise lead to $\mathbf{h}_c \neq \mathbf{0}$ in order to minimize Eq. (4) (ignoring ties).

A.1.3 Equivalence between Eq. (5) and Eq. (6)

Once the closed-form expression of Z is plugged into Eq. (4), the problem can be written as:

$$\begin{aligned} &\min_{H \in \mathcal{Q}^{\mathcal{V}}} \|\operatorname{diag}(H\mathbf{1})XL - HH^\dagger XL\|^2 &(18)\\ =\ &\min_{H \in \mathcal{Q}^{\mathcal{V}}} \operatorname{tr}(\operatorname{diag}(H\mathbf{1})XLL^\top X^\top \operatorname{diag}(H\mathbf{1})) - 2\operatorname{tr}(\operatorname{diag}(H\mathbf{1})XLL^\top X^\top HH^\dagger) + \operatorname{tr}(HH^\dagger XLL^\top X^\top HH^\dagger) &(19)\\ =\ &\min_{H \in \mathcal{Q}^{\mathcal{V}}} \operatorname{tr}(XLL^\top X^\top \operatorname{diag}(H\mathbf{1})\operatorname{diag}(H\mathbf{1})) - 2\operatorname{tr}(XLL^\top X^\top HH^\dagger \operatorname{diag}(H\mathbf{1})) + \operatorname{tr}(XLL^\top X^\top HH^\dagger HH^\dagger) &(20)\\ =\ &\min_{H \in \mathcal{Q}^{\mathcal{V}}} \operatorname{tr}(XLL^\top X^\top \operatorname{diag}(H\mathbf{1})) - 2\operatorname{tr}(XLL^\top X^\top HH^\dagger) + \operatorname{tr}(XLL^\top X^\top HH^\dagger) &(21)\\ \Leftrightarrow\ &\max_{H \in \mathcal{Q}^{\mathcal{V}}} \operatorname{tr}([I - \operatorname{diag}(H\mathbf{1}) + HH^\dagger]XLL^\top X^\top) &(22)\\ =\ &\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XMX^\top \rangle. &(23) \end{aligned}$$

All the matrices in $\mathcal{P}^{\mathcal{V}}$ are orthogonal projection matrices:
The proof in Section A.1.1 implies that, for any $H \in \mathcal{Q}^{\mathcal{V}}$, $[\text{diag}(H\mathbf{1}) - HH^\dagger]$ is an orthogonal projection matrix because:
• it is symmetric (as it is a difference of symmetric matrices).
• it is idempotent by using the proof in Section A.1.1: $[\text{diag}(H\mathbf{1}) - HH^\dagger]^2 = \text{diag}(H\mathbf{1}) + HH^\dagger - HH^\dagger\,\text{diag}(H\mathbf{1}) - \text{diag}(H\mathbf{1})HH^\dagger = \text{diag}(H\mathbf{1}) + HH^\dagger - HH^\dagger - HH^\dagger = \text{diag}(H\mathbf{1}) - HH^\dagger$. Indeed, $\text{diag}(H\mathbf{1})HH^\dagger = ((HH^\dagger)^\top \text{diag}(H\mathbf{1})^\top)^\top = (HH^\dagger\,\text{diag}(H\mathbf{1}))^\top = (HH^\dagger)^\top = HH^\dagger$.
And for all orthogonal projection matrix that is written $P = VDV^\top$ where D is a diagonal matrix whose elements are either 0 or 1 and V is an orthogonal matrix, $I - P = V(I-D)V^\top$ is also an orthogonal projection matrix (as $(I-D)$ is a diagonal matrix whose elements are either 0 or 1). □

A.2. Large margin formulation

Eq. (9) is equivalent to the following large margin problem:

$$\min_{M \in \mathbb{S}^d_+} \max_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \max_{\hat{C} \in f_{M,\mathcal{P}^{\mathcal{G}}}(X)} \Delta(C, \hat{C}) \tag{24}$$

where $\Delta(C,\hat{C}) = n - \langle C, \hat{C} \rangle \geq 0$ measures the *discrepancy* between the two predictions C and $\hat{C}$.

A.3. Proof of Theorem 2.1

We recall that problem (10) is written:

$$\max_{M \in \mathbb{S}^d_+} \min_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C} \rangle \tag{25}$$

Upper bound of Eq. (10): Eq. (10) is naturally upper bounded by

$$\max_{M \in \mathbb{S}^d_+} \max_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C} \rangle \tag{26}$$

By using the definition of $f_{M,\mathcal{P}^{\mathcal{V}}}(X)$ in Eq. (7), we have $f_{M,\mathcal{P}^{\mathcal{V}}}(X) \subseteq \mathcal{P}^{\mathcal{V}}$, Eq. (26) is then upper bounded by:

$$\max_{M \in \mathbb{S}^d_+} \max_{C \in \mathcal{P}^{\mathcal{V}}} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C} \rangle = \max_{C \in \mathcal{P}^{\mathcal{V}}} \max_{M \in \mathbb{S}^d_+} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C} \rangle \tag{27}$$

Let us note $U \in \mathbb{R}^{n \times s}$ a matrix defined as $UU^\top = XX^\dagger$ and $s = \text{rank}(X)$. By using the definition of $g_M(X)$, the column space of $\hat{C}$ is included in the column space of X and $\hat{C}$ is a rank-e orthogonal projection matrix where $e = \text{rank}(XMX^\top) \leq \text{rank}(X) = s$. $\hat{C}$ can then be written: $\hat{C} = UQQ^\top U^\top$ where $Q \in \mathbb{R}^{s \times e}$ and $U \in \mathbb{R}^{n \times s}$ are matrices with orthonormal columns.
Eq. (27) is then upper bounded by:

$$\max_{C \in \mathcal{P}^{\mathcal{V}}} \langle C, UQQ^\top U^\top \rangle = \max_{C \in \mathcal{P}^{\mathcal{V}}} \langle U^\top CU, QQ^\top \rangle \leq \max_{C \in \mathcal{P}^{\mathcal{V}}} \text{tr}(U^\top CU) \tag{28}$$

Indeed, as $Q \in \mathbb{R}^{s \times e}$ is a matrix with orthonormal columns, $\langle U^\top CU, QQ^\top \rangle$ is upper bounded by the sum of the e largest eigenvalues of $U^\top CU$ [21], which is itself upper bounded by $\text{tr}(U^\top CU)$ (as it is the sum of all the eigenvalues of $U^\top CU$ and all the eigenvalues are nonnegative since $U^\top CU$ is symmetric PSD).
Optimal value of Eq. (10): Let us now assume that $M = X^\dagger (X^\dagger)^\top$. In this case, we have the following properties:

$$f_{M,\mathcal{P}^{\mathcal{V}}}(X) = \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XMX^\top \rangle = \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XX^\dagger (X^\dagger)^\top X^\top \rangle = \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XX^\dagger \rangle = \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, UU^\top \rangle \tag{29}$$

$$g_M(X) = \{B : B \in f_{M,\mathcal{N}}(X), \text{rank}(B) \leq \text{rank}(XX^\dagger (X^\dagger)^\top X^\top)\} = \{UU^\top\} \tag{30}$$

The objective value when $M = X^\dagger (X^\dagger)^\top$ is then:

$$\min_{C \in f_{M,\mathcal{P}^{\mathcal{V}}}(X)} \min_{\hat{C} \in g_M(X)} \langle C, \hat{C} \rangle = \min_{C \in \arg\max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, UU^\top \rangle} \langle C, UU^\top \rangle = \max_{C \in \mathcal{P}^{\mathcal{V}}} \text{tr}(U^\top CU) = \max_{A \in \mathcal{P}^{\mathcal{V}}} \langle A, XX^\dagger \rangle \tag{31}$$

The upper bound in Eq. (28) is then obtained, which proves the optimality of the problem for this value. Eq. (11) thus finds an optimal value of C in Eq. (31) (*i.e.* a matrix C that reaches the global optimum value of Eq. (10)). □

A.4. MIL kmeans extension

A.4.1 Why do we optimize Eq. (12)?

We define $U \in \mathbb{R}^{n \times s}$ as a matrix with orthonormal columns such that $s = \operatorname{rank}(X)$ and $XX^\dagger = UU^\top$. U is constructed with the "economy size" singular value decomposition of X and corresponds to the matrix containing the left-singular vectors of the nonzero singular values of X.

By using the results in Section A.1, the problem in Eq. (11) is equivalent to the following problems:

$$\max_{A \in \mathcal{P}^\mathcal{V}} \langle A, XX^\dagger \rangle = \max_{A \in \mathcal{P}^\mathcal{V}} \operatorname{tr}(AXX^\dagger) = \max_{A \in \mathcal{P}^\mathcal{V}} \operatorname{tr}(AUU^\top) = \max_{H \in \mathcal{Q}^\mathcal{V}} \operatorname{tr}([I + HH^\dagger - \operatorname{diag}(H\mathbf{1})]UU^\top) \tag{32}$$

$$\Leftrightarrow \min_{H \in \mathcal{Q}^\mathcal{V}} \operatorname{tr}([\operatorname{diag}(H\mathbf{1}) - HH^\dagger]UU^\top) = \min_{H \in \mathcal{Q}^\mathcal{V}} \operatorname{tr}([\operatorname{diag}(H\mathbf{1}) - HH^\dagger]UU^\top[\operatorname{diag}(H\mathbf{1}) - HH^\dagger]^\top) \tag{33}$$

$$= \min_{H \in \mathcal{Q}^\mathcal{V}} \|[\operatorname{diag}(H\mathbf{1}) - HH^\dagger]U\|^2 \tag{34}$$

$$= \min_{H \in \mathcal{Q}^\mathcal{V}} \|\operatorname{diag}(H\mathbf{1})U - HH^\dagger U\|^2 \tag{35}$$

$$= \min_{H \in \mathcal{Q}^\mathcal{V}, Z \in \mathbb{R}^{k \times s}} \|\operatorname{diag}(H\mathbf{1})U - HZ\|^2 \tag{36}$$

$$= \min_{H \in \mathcal{Q}^\mathcal{V}, Z = [\mathbf{z}_1, \cdots, \mathbf{z}_k]^\top \in \mathbb{R}^{k \times s}} \sum_{j=1}^{n} \sum_{c=1}^{k} H_{jc} \cdot \|\mathbf{u}_j - \mathbf{z}_c\|^2 \text{ where } \mathbf{u}_j^\top \text{ is the } j\text{-th row of } U \tag{37}$$

We then solve Eq. (12) by alternating the optimization over Z and H in Algorithm 1.

A.4.2 Convergence of Algorithm 1

We now prove the convergence of Algorithm 1.

We note $H^{(t)}$ and $Z^{(t)}$ the values at iteration t of $H \in \mathcal{Q}^\mathcal{V}$ and $Z \in \mathbb{R}^{k \times s}$, respectively.

• We first prove that, with Algorithm 1, the sequence of objective values in Eq. (36) (which is equal to Eq. (12)) is monotonically nonincreasing. To this end, we show that:

$$\forall t, \; \|\operatorname{diag}(H^{(t)}\mathbf{1})U - H^{(t)}Z^{(t)}\|^2 \overset{(a)}{\geq} \|\operatorname{diag}(H^{(t)}\mathbf{1})U - H^{(t)}Z^{(t+1)}\|^2 \overset{(b)}{\geq} \|\operatorname{diag}(H^{(t+1)}\mathbf{1})U - H^{(t+1)}Z^{(t+1)}\|^2 \tag{38}$$

- Inequality (a) comes from the fact that $Z^{(t+1)} = (H^{(t)})^\dagger \operatorname{diag}(H^{(t)}\mathbf{1})U = (H^{(t)})^\dagger U$ is a global minimizer of $\min_Z \|\operatorname{diag}(H^{(t)}\mathbf{1})U - H^{(t)}Z\|^2$ as demonstrated in Section A.1.1.

- Inequality (b) comes from the fact that we can decompose the global problem as the sum of m independent subproblems (when the value of Z is fixed):

$$\min_{H \in \mathcal{Q}^\mathcal{V}} \|\operatorname{diag}(H\mathbf{1})U - HZ^{(t+1)}\|^2 = \sum_{i=1}^{m} \min_{H_i \in \mathcal{V}_i} \|\operatorname{diag}(H_i\mathbf{1})U_i - H_iZ^{(t+1)}\|^2 \tag{39}$$

As mentioned in the paper, each subproblem in Eq. (13) is solved exactly with the Hungarian algorithm. The matrix $H^{(t+1)}$ is the concatenation into a single matrix of all the global optimum solutions of the different independent subproblems. It is then a global optimum solution of Eq. (39).

• Our clustering algorithm terminates in a finite number of steps at a partition that is locally optimal (*i.e.*, the total objective value cannot be decreased by either (a) or (b)). This result follows since the sequence of objective values in Eq. (36) is monotonically nonincreasing with Algorithm 1, and the number of distinct clusterings (*i.e.* the cardinality of $\mathcal{P}^\mathcal{V}$, or equivalently the cardinality of $\mathcal{Q}^\mathcal{V}$) is finite. □

A.5. Complexity of Algorithm 1

In the linear case, the complexity of steps 1 and 11 of Algo 1 is dominated by the (economy size) SVDs to compute U and $X^\dagger$ which cost $O(nd\min\{d,n\})$ where d is the dimensionality and n is the number of instances. The adapted kmeans costs $O(r\sum_{i=1}^{m}(sp_iq_i + p_i^2q_i))$ where r is the number of iterations (steps 3 to 8 of Algo 1). Since, in practice, we have $\forall i, \; p_i = \min\{n_i, \mathbf{y}_i^\top\mathbf{1}\} \leq q_i = \max\{n_i, \mathbf{y}_i^\top\mathbf{1}\} \ll n$, the complexity of Algo 1 is dominated by steps 1 and 11 which scale linearly in n as we have $n > d$. In the nonlinear case, computing $K^\dagger J \in \mathbb{R}^{n \times k}$ costs $O(n^3)$; it is efficiently done with a Cholesky solver if K is symmetric positive definite.

In the linear case, the complexity of step 11 of Algorithm 1 does not depend on k and is dominated by the computation of $X^\dagger$ which costs $O(nd \min\{d,n\})$; this is due to the sparsity of H. Indeed, each row of $H \in \{0,1\}^{n \times k}$ contains at most one nonzero element. H then contains at most n nonnzero elements. As explained in Footnote 1, the complexity of computing J such that $JJ^\top = HH^\dagger$ scales linearly in n and J has the same number of nonzero elements as H (*i.e.* at most one per row). Let us note ν_c the number of nonzero elements in the c-th column of J. Once $X^\dagger \in \mathbb{R}^{d \times n}$ has been computed (*i.e.* the value of $X^\dagger$ is known and fixed), computing the c-th row of $X^\dagger J$ costs $O(d\nu_c)$. Computing $L = X^\dagger J$ then costs $O(\sum_{c=1}^k d\nu_c) = O(d \sum_{c=1}^k \nu_c)$. As $\sum_{c=1}^k \nu_c \leq n$, computing $X^\dagger J$ costs $O(dn)$. We actually do not need to compute $M = LL^\top$, computing L is sufficient and then costs $O(nd \min\{d,n\})$ as explained in this section.

A.6. Classification of instances in the nonlinear case

In this section, we extend the classification of test instances in the nonlinear case. To simplify the equations, we assume that the nonlinear kernel function is chosen so that K is invertible (*i.e.*, $K^\dagger = K^{-1}$).

$(\cdot)_{j=1}^n$ denotes concatenation in a n-dimensional vector.

A.6.1 Solving Eq. (15)

The squared distance of a (test) instance $\phi(\mathbf{x}_t)$ to a centroid $\phi(\mathbf{z}_c) = \frac{1}{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}} \Phi \mathbf{h}_c$ where $\mathbf{h}_c \in \{0,1\}^n$ is the c-th column of H is:

$$\begin{aligned}&\|P\Phi^\top \phi(\mathbf{x}_t) - P\Phi^\top \phi(\mathbf{z}_c)\|^2 \\ =&((\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n + ((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n - 2((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n\end{aligned}$$

We recall that $P = J^\top K^{-1}$ and J is defined as explained in Footnote 1, Eq. (15) is then equivalent in the nonlinear case to:

$$\arg\max_{c \in \{1,\cdots,k\}} ((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n - \frac{1}{2}((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n \tag{40}$$

The second (rescaled) term of Eq. (40) can be written:

$$\begin{aligned}((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n &= \frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c^\top \Phi^\top \Phi K^{-1} J J^\top K^{-1} \Phi^\top \Phi(\frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c) && (41)\\ &= \frac{1}{(\max\{1,\mathbf{h}_c^\top\mathbf{1}\})^2}\mathbf{h}_c^\top K K^{-1} J J^\top K^{-1} K \mathbf{h}_c && (42)\\ &= \frac{1}{(\max\{1,\mathbf{h}_c^\top\mathbf{1}\})^2}\mathbf{h}_c^\top J J^\top \mathbf{h}_c = \frac{1}{(\max\{1,\mathbf{h}_c^\top\mathbf{1}\})^2}\mathbf{h}_c^\top H H^\dagger \mathbf{h}_c && (43)\\ &= \frac{1}{(\max\{1,\mathbf{h}_c^\top\mathbf{1}\})^2}\mathbf{h}_c^\top \mathbf{h}_c && (44)\end{aligned}$$

We also note that $\|\mathbf{h}_c\|^2 = \mathbf{h}_c^\top \mathbf{h}_c = \mathbf{h}_c^\top \mathbf{1} = \sum_j H_{jc}$ is the number of instances assigned to category c. Eq. (44) is then equal to the inverse of the number of elements assigned to category c (*i.e.* the inverse of the size of cluster c) if $\mathbf{h}_c \neq \mathbf{0}$, and 0 otherwise.

The first term of Eq. (40) can be written:

$$\begin{aligned}((\mathsf{k}(\mathbf{x}_j,\mathbf{z}_c))_{j=1}^n)^\top P^\top P\,(\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n &= \frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c^\top \Phi^\top \Phi K^{-1} J J^\top K^{-1} (\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n && (45)\\ &= \frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c^\top K K^{-1} J J^\top K^{-1} (\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n && (46)\\ &= \frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c^\top H H^\dagger K^{-1} (\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n && (47)\\ &= \frac{1}{\max\{1,\mathbf{h}_c^\top\mathbf{1}\}}\mathbf{h}_c^\top K^{-1} (\mathsf{k}(\mathbf{x}_j,\mathbf{x}_t))_{j=1}^n && (48)\end{aligned}$$

Number of instances in a bag	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Number of bags	12562	5109	1675	480	146	61	17	8	6	0	1	3	1	1	1

Table 5. Distribution of the number of instances per bag: 12562 bags contain one instance, 5109 bags contain 2 instances etc.

Number of training categories in bags	0	1	2	3	4	5	6	7	8	9
Scenario (b)	1384	16196	2295	181	8	3	1	2	0	1
Scenario (c)	0	12225	6247	1325	216	46	8	3	0	1

Table 6. Distribution of the number of training categories (*i.e.*, among the $k = 5873$) labeled as present in the bags depending on the scenarios. 1384 bags contain 0 training category in scenario (b) as instances correspond to other persons or are not face instances. etc.

Scenario	Evaluation	M-C2B [29]	miSVM [1]	MILES [4]	MILBoost [36]	EM-DD [37]	Minimax MI-Kernel [10]	MinD *(minmin)* [5]	MinD *(maxmin)*	MinD *(meanmin)*
(b)	Accuracy (%)	6.6 ± 2.2	4.5 ± 2.7	8.2 ± 2.3	8.8 ± 2.4	1.3 ± 0.5	5.5 ± 1.7	6.8 ± 2.5	3.2 ± 1.5	5.1 ± 1.9
	Precision (%)	7.2 ± 2.5	2.3 ± 1.5	9.2 ± 2.7	9.7 ± 2.7	1.8 ± 0.8	6.2 ± 2.5	7.1 ± 2.4	3.1 ± 1.4	5.5 ± 1.8
	Train. Time (s)	$2,572$	610	240	182	$13,163$	358	276	259	265
(c)	Accuracy (%)	4.5 ± 1.8	3.6 ± 1.2	6.7 ± 2.0	6.9 ± 2.3	0.8 ± 0.2	4.8 ± 1.0	5.5 ± 1.3	1.8 ± 1.0	3.6 ± 1.2
	Precision (%)	5.3 ± 1.9	1.5 ± 0.8	7.0 ± 1.2	7.6 ± 1.8	1.1 ± 0.3	4.6 ± 1.3	5.3 ± 1.3	1.5 ± 0.7	3.4 ± 0.8
	Train. Time (s)	$2,762$	653	265	205	$13,484$	391	296	281	291

Table 7. Performance of the different baselines on the *Labeled Yahoo! News* dataset.

A.6.2 Solving Eq. (16)

Following Section A.6.1, Eq. (16) can be adapted in the following way:

$$\arg\max_{c \in \{1,\cdots,k\}} \frac{1}{\sqrt{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}}} \mathbf{h}_c^\top K^{-1} (\mathsf{k}(\mathbf{x}_j, \mathbf{x}_t))_{j=1}^n - \frac{\alpha}{(\max\{1, \mathbf{h}_c^\top \mathbf{1}\})^2} \mathbf{h}_c^\top \mathbf{h}_c \tag{49}$$

$$\Leftrightarrow \arg\max_{c \in \{1,\cdots,k\}} \mathbf{j}_c^\top K^{-1} (\mathsf{k}(\mathbf{x}_j, \mathbf{x}_t))_{j=1}^n - \frac{\alpha}{(\max\{1, \mathbf{h}_c^\top \mathbf{1}\})^2} \mathbf{h}_c^\top \mathbf{h}_c \tag{50}$$

where $\mathbf{j}_c = \frac{1}{\sqrt{\max\{1, \mathbf{h}_c^\top \mathbf{1}\}}} \mathbf{h}_c$ is the c-th column of J as explained in Footnote 1.

A.7. Statistics of Labeled Yahoo News! dataset

We give some statistics of the Labeled Yahoo News! dataset in Tables 5 and 6.

A.8. Scores of biclass MIL classifiers

Baselines results are reported in Table 7. As M-C2B [29] uses an iterative algorithm and the complexity of each of its iterations is cubic in d, we had to reduce the dimensionality to $d = 1000$ via PCA to make it scalable.

As explained in Section 3, M-C2B [29] is not appropriate for the face recognition task as it considers that all the instances in bags that contain a given category are relevant to the category. In the case of face verification, at most one instance per bag is relevant to a given category.

A.9. Interpretation of the results of MIMLCA on *Labeled Yahoo! News*

On test categories (*i.e.*, the ~ 50 selected categories per split), our model actually finds the correct instance assignments of training instances with an error of 8.6% in scenario (b) and 16.2% in scenario (c); the larger the number of instances in the categories, the smaller the detection error.

A.10. Our reimplementation of [12]

We contacted in April 2016 the authors of [12] and asked for their code. They replied that their code was not available. Here is our reimplementation of their method:

```
function [A, Z, Obj] = MIML_metric(X, Y, N, r, params)
% X : [N_1, N_2, ...] in R^{d x t}
% Y : bool valued in {0,1}^{n x m}
%         d: feature dimension
%         n : number of bags
%         m : number of labels
%         t : total number of instances
% N : n x 1, N(ii) is the number of instances in bag ii
%         for equal sized bags, N can be 1 x 1
% r : reduced dimension of the metric
% params : parameters, structure
%           params.iter, max outer iteration
%           params.inner, max inner iteration
%           params.TOL, tolerance
%
% A : AA' is the distance metric, A orthogonal
%         in R^{d x r}
% Z : centroids, in R^{d x m}
%         each class has only one centroid (as in the experiments of Rong Jin et al.)

[d, t] = size(X);
[n, m] = size(Y);

% convenience for equal size of bags
if length(N) == 1, N = repmat(N, n, 1); end
if nargin < 4
    error('not enough inputs');
elseif nargin == 4
    params = [];
end
if isempty(params)
    params.iter = 50;
    params.inner = 20;
    params.TOL = 1e-4;
end
max_iter = params.iter;
max_inner = params.inner;
TOL = params.TOL;

% initialize Mahalanobis metric
[A, ¬] = qr(randn(d, r), 0);
% initialize the centers;
%       each class has one center (as in the experiments of Rong Jin et al.)
Z = randn(d, m);
% initialize Q
Q = zeros(n, m);
Obj = zeros(max_iter, 1);
for iter = 1:max_iter

    % Optimizing Q with A and Z fixed
    Xhat = A' * X;
    Zhat = A' * Z;
    Sim = Xhat' * Zhat;
    LenX = sum(Xhat.^2, 1)'; % COL
    LenZ = sum(Zhat.^2, 1); % ROW
    % (squared) distance between X and Z: t x m
    Dist = repmat(LenX,1,m) - 2*Sim + repmat(LenZ,t,1);

    % find Q bag by bag
    cum = 0;
    for ii = 1:n
        [¬, Q(ii,:)] = min(Dist(cum+1:cum+N(ii), :), [], 1);
        % fix the index
        Q(ii, :) = Q(ii, :) + cum;
        cum = cum + N(ii);
    end

    % Optimizing A with Q and Z fixed
    % forming U by replication
    Xsel = X(:, Q(:)); % [n n ... n]
    Zrep = repelem(Z, 1, n); % [n n ... n]
    U = (Xsel - Zrep) * diag(Y(:)) * (Xsel - Zrep)';
    % forming V by Laplacian
    V = 2 * Z * (m*eye(m) - ones(m)) * Z';
    % generalized eigen-decomposition

    %% debug
    %     Diff = A'*Xsel - repelem(A'*Z, 1, n);
    %     obj = sum(Diff.^2, 1) * Y(:);
    %%
    sigma = 0;
    for ii = 1:max_inner
        D = V - sigma*U;
        D = (D+D') / 2;
        [A, ¬] = eigs(D, r, 'LA');
        sigma_new = trace(A'*V*A) / (trace(A'*U*A)+eps);
        if abs(sigma_new - sigma) ≤ sigma*TOL
            break;
        end
        sigma = sigma_new;
        %% debug
        %         Diff = A'*Xsel - repelem(A'*Z, 1, n);
        %         obj = sum(Diff.^2, 1) * Y(:);
        %%
    end


    % Optimizing Z with Q and A fixed
    Xhat = A' * Xsel;
    Zhat = A' * Z;

    % maintain some invariants
    sumZ = sum(Zhat, 2);
    InnerProd = Zhat' * Zhat;
    sqNormZ = trace(InnerProd);
    simZ = sum(InnerProd(:));

    tmp = Xhat .* repmat(Y(:)', r, 1);
    tmp = reshape(tmp, r, n, m);
    % not to confuse with V
    VV = squeeze(sum(tmp, 2));

    %% h is not needed
    % sqNormX = sum(Xhat.^2, 1);
    % sqNormX = repmat(sqNormX, n, m);
    % h = sum(sqNormX.*Y, 1);

    % not to confuse with A
    AA = sum(Y, 1);

    % not to confuse with t, total number of instances
    tfix = trace(Zhat * ((m+1)*eye(m) - ones(m)) * Zhat') / 2;

    Diff = Xhat - repelem(Zhat, 1, n);
    obj = sum(Diff.^2, 1) * Y(:);
    for ii = 1:max_inner
        for jj = 1:m
            z = Zhat(:, jj);
            u = (sumZ - z) / (m-1);
            s = (tfix - m*sqNormZ + (m+1)*(z'*z) + simZ - 2*z'*sumZ) / (m-1);
            a = AA(jj);
            v = VV(:, jj);

            den = s + norm(u)^2;
            if den > 0
                lambda = a - min(a, norm(v-a*u)/sqrt(den));
            else
```

```
                lambda = 0;
            end
            znew = (v-lambda*u) / (a-lambda);

            Zhat(:, jj) = znew;

            % update the invariants
            simZ = simZ - 2*z'*sumZ;
            sumZ = sumZ - z + znew;
            sqNormZ = sqNormZ - z'*z + znew'*znew;
            simZ = simZ + 2*znew'*sumZ;
        end

        Diff = Xhat - repelem(Zhat, 1, n);
        obj_new = sum(Diff.^2, 1) * Y(:);
        if  abs(obj - obj_new) ≤ TOL*obj_new
            break; % converged
        end
        obj = obj_new;
    end

    fprintf('iter = %d, obj = %f \n', iter, obj);
    if iter > 1 && abs(Obj(iter-1) - obj) ≤ TOL*obj
        break; % converged
    end

    Obj(iter) = obj;

    % recover Z in full dimension
    Z = A * Zhat;
end
Obj = Obj(1:iter);
```

A.11. Reimplementation of [29]

The reimplementation of [29, Algorithm 1] is straightforward. We use the same variable names as in the original paper:

```
function [ L, tElapsed ] = robust_mil(U,A,B, max_nbiterations, epsilon)
best_obj = inf;
obj = inf;
tStart = tic;
for iter=1:max_nbiterations
    % step 2: construct lambda
    lambda = sum(sqrt(sum((A * U).^2,2))) /  sum(sqrt(sum((B * U).^2,2)));
    % step 3: construct D
    D = diag(1 ./ (2 * sqrt(sum((A * U).^2,2))));
    % step 4: construct S
    bU = (B * U)';
    norm_bU = sqrt(sum(bU.^2,1));
    S = (bsxfun(@rdivide,bU,norm_bU))';
    % we use pinv instead of the operator \ because 2*(A'*D*A) is sometimes ill-conditioned
    U = lambda *  pinv(2 * (A' * D * A)) * (B' * S);
    old_obj = obj;
    obj = trace(U'*A'*D*A*U) - lambda * trace(U'*B'*S);
    if obj ≤ best_obj
        best_obj = obj;
        best_U = U;
    end
    if abs(old_obj - obj) < epsilon
        break;
    end
end
tElapsed = toc(tStart)
L = best_U;
end
```