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 $H \in \mathcal{Q}^{\mathcal{V}}$, $[\operatorname{diag}(H\mathbf{1}) - HH^\dagger]$ 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: $[\operatorname{diag}(H\mathbf{1}) - HH^\dagger]^2 = \operatorname{diag}(H\mathbf{1}) + HH^\dagger - HH^\dagger \operatorname{diag}(H\mathbf{1}) - \operatorname{diag}(H\mathbf{1})HH^\dagger = \operatorname{diag}(H\mathbf{1}) + HH^\dagger - HH^\dagger - HH^\dagger = \operatorname{diag}(H\mathbf{1}) - HH^\dagger$. Indeed, $\operatorname{diag}(H\mathbf{1})HH^\dagger = ((HH^\dagger)^\top \operatorname{diag}(H\mathbf{1})^\top)^\top = (HH^\dagger \operatorname{diag}(H\mathbf{1}))^\top = (HH^\dagger)^\top = HH^\dagger$.
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 of

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 $f_{M,\mathcal{P}^{\mathcal{V}}}(X)$ 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 $s = \operatorname{rank}(X)$. 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 $e = \operatorname{rank}(XMX^\top) \leq \operatorname{rank}(X) = s$. $\hat{C}$ 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:

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

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 $\operatorname{tr}(U^\top CU)$ (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:

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

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

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

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

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 $s = \text{rank}(X)$ 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:

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

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

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

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

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

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

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:

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

- Inequality (a) comes from the fact that $Z^{(t+1)} = (H^{(t)})^\dagger \text{diag}(H^{(t)}\mathbf{1})U = (H^{(t)})^\dagger U$ is a global minimizer of $\min_Z \|\text{diag}(H^{(t)}\mathbf{1})U - H^{(t)}Z\|^2$ 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):

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

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. $\square$

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)*
	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
(b)	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
	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
(c)	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
```