Supplemental Material for
What Makes Objects Similar:
A Unified Multi-Metric Learning Approach

Han-Jia Ye, De-Chuan Zhan, Yuan Jiang, and Zhi-Hua Zhou, Fellow, IEEE

Abstract—This is the supplemental material for the manuscript “What makes objects similar: A unified multi-metric learning approach”. We first present the detailed optimization process of U\(M^2L\), when we need to learn the threshold value. Then more explanations and experimental results are presented, namely, implementation details, feature pattern discovery, ablation investigation, and multi-view detection. Last, we discuss the proposed reweighted solver for the symmetric \(\ell_{2,1}\)-norm and its convergence.

Index Terms—Distance Metric Learning, Multi-Metric Learning, Similarity measures, Semantic

1 UM\(M^2L\) THRESHOLD OPTIMIZATION

Multiple metrics \(\mathcal{M}_K = \{M_1, \ldots, M_K\}\) could be used to explain various semantic meanings between objects. For (squared) Mahalanobis distance between a pair \((x_i, x_j)\) with metric \(M_k\), i.e.,

\[
\text{Dis}^2_{M_k}(x_i, x_j) = (x_i - x_j)\top M_k (x_i - x_j) = \text{Tr}(M_k A_{i,j}),
\]

(1)

the overall similarity between \(x_i\) and \(x_j\), could be defined as:

\[
f_{q_{ij}}(x_i, x_j) = \kappa_{q_{ij}} (-\text{Dis}^2_{M_k}(x_i, x_j), \ldots, -\text{Dis}^2_{M_K}(x_i, x_j)) .
\]

\(\kappa_{q_{ij}}\) is an operator combining/selecting \(K\) similarity values (negative distances) together. Given totally \(P\) pairwise side information \(\mathcal{P} = \{(x_i, x_j, q_{ij})\}\), and the value \(q_{ij} \in \{-1, 1\}\) reveals whether two instances are similar or not. The pairwise version of our Unified Multi-Metric Learning framework can be formalized as

\[
\min_{\mathcal{M}_K} \frac{1}{P} \sum_{(i,j) \in \mathcal{P}} \ell(q_{ij}(f_{q_{ij}}(x_i, x_j) - \gamma)) + \lambda \sum_{k=1}^{K} \Omega_k(M_k) .
\]

(2)

In Eq.\(2\), \((i, j) \in \mathcal{P}\) enumerate all pairs in \(\mathcal{P}\); \(\ell(\cdot)\) is a loss function, the lower the better; \(\Omega(\cdot)\) is the metric regularizer; and \(\gamma > 0\) is a threshold value.

Using the objective in Eq.\(2\) the overall similarity values between similar objects should be larger than \(\gamma\); while for a dissimilar pair, their similarity value should be small. The value of \(\gamma\) could be fixed in advance, or learned in the training process.

As described in the manuscript, UM\(M^2L\) can be solved alternatively between metrics \(\mathcal{M}_K\) and affiliation portion of each instance, when \(\kappa_{q_{ij}}\) are piecewise linear operators such as \(\max(\cdot)\) and \(\min(\cdot)\). Specifically, given current learned \(\mathcal{M}_K\), the metric used to measure the similarity of pair \(\tau = (x_i, x_j)\)

\[\underbrace{\kappa_{\tau}}_{\gamma} \text{version of our Unified Multi-Metric Learning framework can be formalized as:} \quad \text{Dis}^2_{M_k}(x_i, x_j) = (x_i - x_j)\top M_k (x_i - x_j) = \text{Tr}(M_k A_{i,j}), \]

\[f_{q_{ij}}(x_i, x_j) = \kappa_{q_{ij}} (-\text{Dis}^2_{M_k}(x_i, x_j), \ldots, -\text{Dis}^2_{M_K}(x_i, x_j)) .
\]

\(\kappa_{q_{ij}}\) is an operator combining/selecting \(K\) similarity values (negative distances) together. Given totally \(P\) pairwise side information \(\mathcal{P} = \{(x_i, x_j, q_{ij})\}\), and the value \(q_{ij} \in \{-1, 1\}\) reveals whether two instances are similar or not. The pairwise version of our Unified Multi-Metric Learning framework can be formalized as

\[
\min_{\mathcal{M}_K} \frac{1}{P} \sum_{(i,j) \in \mathcal{P}} \ell(q_{ij}(f_{q_{ij}}(x_i, x_j) - \gamma)) + \lambda \sum_{k=1}^{K} \Omega_k(M_k) .
\]

(2)

In Eq.\(2\), \((i, j) \in \mathcal{P}\) enumerate all pairs in \(\mathcal{P}\); \(\ell(\cdot)\) is a loss function, the lower the better; \(\Omega(\cdot)\) is the metric regularizer; and \(\gamma > 0\) is a threshold value.

Using the objective in Eq.\(2\) the overall similarity values between similar objects should be larger than \(\gamma\); while for a dissimilar pair, their similarity value should be small. The value of \(\gamma\) could be fixed in advance, or learned in the training process.

As described in the manuscript, UM\(M^2L\) can be solved alternatively between metrics \(\mathcal{M}_K\) and affiliation portion of each instance, when \(\kappa_{q_{ij}}\) are piecewise linear operators such as \(\max(\cdot)\) and \(\min(\cdot)\). Specifically, given current learned \(\mathcal{M}_K\), the metric used to measure the similarity of pair \(\tau = (x_i, x_j)\) can be determined directly. For example, if \(\kappa = \max(\cdot)\), then \(k^*_\tau = \arg \max_k \text{Dis}_k(x_i, x_j)\), which is the index of the metric in \(\mathcal{M}_k\) that has the largest similarity value over the pair. Once the dominating key metric of each instance is found, the whole optimization problem considers only one active metric for each pair, which can be easily optimized.

It is notable that in this sub-problem, we can optimize over threshold \(\gamma\) at the same time.

With the smooth hinge loss, the gradient of loss function w.r.t. threshold parameter \(\gamma\) can be computed by

\[
\frac{\partial \ell(M_k)}{\partial \gamma} = \frac{1}{P} \sum_{(i,j) \in \mathcal{P}} \frac{\partial \ell(q_{ij}(\text{Dis}_\tau(x_i, x_j) - \gamma))}{\partial \gamma},
\]

(3)

The gradient value \(\nabla_{M_k}(a_{\tau}) = 0\) when \(a_{\tau} \geq 1\); \(\nabla_{M_k}(a_{\tau}) = (1 - a_{\tau})q_{ij}\delta[k^*_\tau = k]\) when \(0 < a_{\tau} < 1\); and \(\nabla_{M_k}(a_{\tau}) = q_{ij}\delta[k^*_\tau = k]\) when \(0 < a_{\tau} \leq 0\). The gradient computation of \(\gamma\) can be used in the accelerated projected gradient descent or the accelerated proximal gradient descent when solving UM\(M^2L\) variants.

The learned \(\gamma\) helps the determination of similar relationship between two objects, e.g., in the social linkage discovering experiment.

2 IMPLEMENTATION DETAILS

When optimizing UM\(M^2L\) in an alternative style, we initialize the metric affiliation portion for each instance first. The affiliation portion depends on which metric in \(\mathcal{M}_K\) to use. GMM or KMeans is conducted with the component number the same as \(K\) set in UM\(M^2L\). After that, each instance can be represented as a vector of length \(K\). Comparison between these coding values indicates which component to illustrate the similar/dissimilar relationship. For instance, in Apical Dominating Similarity (ADS) with \(\kappa_1 = \kappa_2 = \max(\cdot)\), given
when both of their affiliations are not small w.r.t. a certain Vote Similarity (OVS) and Rank Grouping Similarity (RGS),

To demonstrate the ability of linkage decomposition, we select that the whole objective is convex w.r.t. dissimilar

samplered from Uniform, Beta, Binomial, Gamma and Normal

rows/columns.

\[ \ell \]

diverse among different views. After that, the overall linkage

is further generated from these candidates using an “or” operation. It is notable that the linkage generation process is

are selected based on Euclidean nearest neighbors to generate pairs. While for the triplet version, we use 3 target neighbors and 10 impostors for each instance as in the usual setting [4]. [5]. The component number \( K \) is set the same as the number of classes. \( \ell^1 \)-norm is used as the regularizer. For each dataset, the evaluations are repeated for 30 times. In each trial, 70% of instances are used for training, and the remaining part is for the test. For all comparison methods, we select the parameter \( \lambda \) using cross-validation. Performances of different \( \lambda \) are computed in the training set, and the \( \lambda \) with the best performance is used to train the whole training set and for the test. 3NN results (error mean ± std.) are listed in Table 2.

From Table 2, the triplet versions of \( \mathbb{U}L^2 \) get better and more stable results in most cases, it may be the reason that more local information is utilized during training compared with the pairwise versions. OVS could not output comparable results in some cases, and sometimes the performance is not stable, since the explain of linkages in OVS is more for a decomposition goal, not for classification. It is able to find the possible semantic component in the object as shown in other subsections.

4 Full results of Ablation Investigation

Before comparing \( \mathbb{U}L^2 \) with others, we first analyze the classification performance of \( \mathbb{U}L^2 \) variants: for different selection of \( \kappa \), namely, ADS/OVS/RGS, and both the pairwise/triplet versions. The summation form \( \kappa = \sum \) is also listed as a baseline. Because in \( \mathbb{U}L^2 \) distance values from different metrics are comparable, so in the test phase, a variant of \( kNN \) is applied to use multiple learned metrics. With 3NN, we first compute 3 nearest neighbors for testing instance \( \tilde{x} \) using each base metric \( M_k \). Then \( 3 \times K \) distance values are collected adaptively, and the smallest three (with the highest similarity scores) form neighbor candidates. Majority voting over them is used for prediction.

In the pairwise implementation, 10 targets and impostors are selected based on Euclidean nearest neighbors to generate pairs. For the triplet version, we use 3 target neighbors and 10 impostors for each instance as in the usual setting [4]. [5]. The component number \( K \) is set the same as the number of classes. \( \ell^1 \)-norm is used as the regularizer. For each dataset, the evaluations are repeated for 30 times. In each trial, 70% of instances are used for training, and the remaining part is for the test. For all comparison methods, the selection of \( \lambda \) using cross-validation. Performances of different \( \lambda \) are computed in the training set, and the \( \lambda \) with the best performance is used to train the whole training set and for the test. 3NN results (error mean ± std.) are listed in Table 2.

From Table 2, the triplet versions of \( \mathbb{U}L^2 \) get better and more stable results in most cases, it may be the reason that more local information is utilized during training compared with the pairwise versions. OVS could not output comparable results in some cases, and sometimes the performance is not stable, since the explain of linkages in OVS is more for a decomposition goal, not for classification. It is able to find the possible semantic component in the object as shown in other subsections.

5 Investigations of Latent Multi-View Detection

Another direct application of \( \mathbb{U}L^2 \) is hidden multi-view detection, where data can be described by multiple views from different channels yet feature partitions are not clearly provided [6]. The hidden multi-view data [7] are composed of 200 instances and each instance has two hidden views, namely the color and the shape. We perform \( \mathbb{U}L^2\text{LADS/OVS/RGS} \) on this dataset with \( K = 2 \). Trace norm regularizes the approach in this part to get low dimensional projections. \( \mathbb{U}L^2 \) framework facilitates the understanding of data by decomposing each base metric to low dimensional subspace projector, i.e., for each base metric \( M_k \), two eigenvectors \( L_k \in \mathbb{R}^{d \times 2} \) corresponding to the largest 2 eigenvalues are picked. The results can be found in Fig. 1. We use

<table>
<thead>
<tr>
<th>BER ( \mathbb{U}L^2 )</th>
<th>KM</th>
<th>SP</th>
<th>SCA</th>
<th>EGO</th>
<th>( \mathbb{U}L^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>syn1</td>
<td>0.382</td>
<td>0.382</td>
<td>0.392</td>
<td>0.467</td>
<td>0.355</td>
</tr>
<tr>
<td>syn2</td>
<td>0.564</td>
<td>0.564</td>
<td>0.399</td>
<td>0.428</td>
<td>0.323</td>
</tr>
<tr>
<td>ad</td>
<td>0.670</td>
<td>0.670</td>
<td>0.400</td>
<td>0.583</td>
<td>0.381</td>
</tr>
<tr>
<td>ccd</td>
<td>0.244</td>
<td>0.244</td>
<td>0.250</td>
<td>0.225</td>
<td>0.071</td>
</tr>
<tr>
<td>my_movie</td>
<td>0.370</td>
<td>0.370</td>
<td>0.249</td>
<td>0.347</td>
<td>0.155</td>
</tr>
<tr>
<td>reuters</td>
<td>0.704</td>
<td>0.704</td>
<td>0.400</td>
<td>0.609</td>
<td>0.398</td>
</tr>
</tbody>
</table>

Table 1: BER (the lower the better) of feature pattern discovery comparisons on synthetic datasets: \( \mathbb{U}L^2 \text{LADS} \) vs. others

similar pair \((x_i, x_j)\) with coding vector \((\tilde{x}_i, \tilde{x}_j)\) in \( \mathbb{R}^K \), the initial metric of this pair is selected by:

\[ k = \arg \max_k \min_i \min_j (\tilde{x}_i, \tilde{x}_j). \]

The selection can be interpreted as a two-stage process. First, similarities between instances based on \( K \) components are calculated in a Histogram Intersection Kernel (HIK) style, where instances are similar based on some metric only when both of their affiliations are not small w.r.t. a certain component. Second, a particular metric is selected based on the largest pair-similarity value.

There are three types of similarity defined in the main body, namely Apical Dominating Similarity (ADS), One Vote Similarity (OVS) and Rank Grouping Similarity (RGS), which have different optimization properties. The RGS is convex, and thus alternative approach can get global optimal solutions. The OVS is a non-convex one, whose results depend on initializations. The ADS is a semi-convex problem [4]. To better utilize the property of the semi-convex part that the whole objective is convex w.r.t. dissimilar pairs, we compute only the affiliation of similar pairs in the initialization phase. Since when these affiliations are fixed, the whole problem becomes a convex one, even there still exists a \( \max(\cdot) \) operator.

3 Feature Pattern Discovering

To demonstrate the ability of linkage decomposition, we test feature pattern discovering the ability of \( \mathbb{U}L^2 \text{LADS} \) on 4 transformed multi-view datasets [2]. For each dataset, we first extract principal components of each view, and construct sub-linkage candidates between instances with random thresholds on each single view. Thus, these candidates are diverse among different views. After that, the overall linkage is further generated from these candidates using an “or” operation. It is notable that the linkage generation process is the same as the property of similarity linkages for \( \mathbb{U}L^2 \text{LADS} \). With features on each view and the overall linkage, the goal of feature pattern discovering is to reveal responsible features for each sub-linkage.

With multiple learned metrics, the zero-valued rows and columns of the learned metrics indicate irrelevant features in the corresponding group. It is both the intrinsic feature sparsity in linkage generation and the sparse property of \( \ell^1 \)-norm makes the final multiple metrics sparse over rows/columns.

Syn1 and syn2 are purely synthetic datasets with features sampled from Uniform, Beta, Binomial, Gamma and Normal distributions using two sets of different parameters. Balanced Error Rate (BER) [3] between ground truth feature indicator sets and predicted sets are listed in Table 1. \( \mathbb{U}L^2 \text{LADS} \) achieves the best on all datasets. These assessments indicate \( \mathbb{U}L^2 \text{LADS} \) can figure out reasonable linkages or patterns hidden behind observations, and even better than domain specific methods.
V1 and V2 to denote the projection results of a certain view. NV1 and NV2 show the case when we combine original features with random noise.

We can explain the linkages between objects among views using different Um²L variants. ADS emphasizes the existence of relevant views and aims at decomposing helpful aspects or views; different from ADS, the OVS permits the reason for dissimilar objects also in a certain view; while RGS requires full accordance among views, the dissimilar and similar relationship should be kept in both views.

We set the parameter $\lambda$ here as default value 1. When we increase the parameter, Um²L is able to generate more sparse views (more information in a smaller number of views). For example, as in plot (g), there is only a single view for RGS output, which covers both color and shape semantics in one plot (although set to learn 2 metrics at first, the other one degenerate to zero after training).

From the results, it is also notable that ADS could discover two semantics even in noisy cases. The results of RGS are more discriminative than ADS. OVS outputs the same projections as RGS, but only one output in the noisy case. It could be the reason that OVS only requires a “weak” type of similarity generation. So when facing the noise, it concentrates the two semantics into one single view.

<table>
<thead>
<tr>
<th>$\kappa$</th>
<th>Pairwise</th>
<th>Tripet</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADS</td>
<td>OVS</td>
<td>RGS</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Automp</td>
<td>205 ± 0.06</td>
<td>0.377 ± 0.046</td>
</tr>
<tr>
<td>Clean1</td>
<td>1.02</td>
<td>0.02</td>
</tr>
<tr>
<td>German</td>
<td>277 ± 0.016</td>
<td>0.326 ± 0.102</td>
</tr>
<tr>
<td>Glass</td>
<td>0.356</td>
<td>0.054</td>
</tr>
<tr>
<td>Hayes-r</td>
<td>0.327</td>
<td>0.044</td>
</tr>
<tr>
<td>Heart-s</td>
<td>0.206</td>
<td>0.038</td>
</tr>
<tr>
<td>House-v</td>
<td>0.046 ± 0.017</td>
<td>0.103</td>
</tr>
<tr>
<td>Liver-d</td>
<td>0.393</td>
<td>0.037</td>
</tr>
<tr>
<td>Segment</td>
<td>0.022 ± 0.005</td>
<td>0.068</td>
</tr>
<tr>
<td>Sonar</td>
<td>0.167 ± 0.040</td>
<td>0.201</td>
</tr>
</tbody>
</table>

Table 2: Test error comparison between different $\kappa$ implementations in the Um²L framework, with both pairwise and triplet variants.

<table>
<thead>
<tr>
<th></th>
<th>MVTE</th>
<th>mmTSNE</th>
<th>SCA</th>
<th>ADS</th>
<th>OVS</th>
<th>RGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Noise</td>
<td>View1</td>
<td>0.960</td>
<td>0.929</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>View2</td>
<td>0.833</td>
<td>0.29</td>
<td>0.919</td>
<td>1</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Mean</td>
<td>0.896</td>
<td>0.479</td>
<td>0.959</td>
<td>1</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Noise</td>
<td>View1</td>
<td>N/A</td>
<td>N/A</td>
<td>0.764</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>View2</td>
<td>N/A</td>
<td>N/A</td>
<td>0.503</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Mean</td>
<td>N/A</td>
<td>N/A</td>
<td>0.633</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: The Paired Semantic NMI (PSNMI) results for all comparison methods, for original and noisy cases. We use ADS/OVS/RGS to denote the case when using different $\kappa$ in Um²L. “N/A” in the table shows there is no applied result.
6 Reweighted Method on Symmetric $\ell_{2,1}$-norm

Solving $\ell_{2,1}$-norm under symmetric constrained can be transformed to the proximal sub-problem in

$$M_k' = \arg \min_{M \in S_d} \frac{1}{2} \| M - V_k \|_F^2 + \lambda_1 \| M \|_{2,1} + \lambda_2 \| M^T \|_{2,1}. \quad (4)$$

We ignore the subscript index $k$ in the following discussions. Since $\ell_{2,1}$-norm is the sum of $\ell_2$-norm on each row of $M$, which violates the symmetric property, directly optimizing may be time-consuming in some cases [8]. Taking symmetry into consideration, we reformulate the problem in Eq. (4) as follows:

$$M = \arg \min_{M \in S_d} \frac{1}{2} \| M - V \|_F^2 + \lambda_1 \| M \|_{2,1} + \lambda_2 \| M^T \|_{2,1}. \quad (5)$$

The impact of $\ell_{2,1}$-norm is shared on $M$ and $M^T$ equally. Taking derivative of Eq. (5) w.r.t. $M$ and setting it to zero get:

$$M - V + \lambda_1 D_1 M + \lambda_2 D_2 M^T = 0. \quad (6)$$

Both $D_1$ and $D_2$ are diagonal matrices of size $d \times d$, and the $(r,c)$-th elements in $D_1$ and $D_2$ are $a_{1}^{r} = \frac{1}{\|m_{r,c}\|^2}$ and $a_{2}^{r,c} = \frac{1}{\|m_{r,c}\|^2}$, respectively. Thus, $D_1$ and $D_2$ consider $\ell_2$-norm values from both rows and columns of $M$. Eq. (5) is a Sylvester equation, which has a high computational cost when solved with off-the-shelf tools. To accelerate, we consider the closed form results based on Kronecker product $\otimes$, and comes to the result of Lemma 1:

$$\text{vec}(M) = (I \otimes (I + \frac{\lambda_1}{2} D_1)) + (\frac{\lambda_2}{2} D_2 \otimes I))^{-1} \text{vec}(V). \quad (7)$$

$\text{vec}(\cdot)$ means the vectorization of a matrix. It is notable that all terms, i.e., $D_1$, $D_2$ and identity matrix $I$, in the inverse operation are in diagonal forms. So the inverse is not on a matrix but on a scalar. The $(r,c)$-element in $M$ is:

$$M_{rc} = V_{rc}/((1 + \frac{\lambda_1}{2} a_{1}^{r}) + \frac{\lambda_2}{2} a_{2}^{r,c}). \quad (8)$$

Since metric $M$ keeps symmetry in each iteration, this update flow is symmetric projection free. Thus, $a_{1}^{r} = a_{1}^{c}$, which further simplifies the computation. Since $D_1$ and $D_2$ in the closed form update of $M$ in Eq. (8) also depend on $M$, they should be updated alternatively. The convergence of this reweighted method can be easily proved [9] and is validated in our experiments in section 7.5.

6.1 Convergence of Solving Symmetric $\ell_{2,1}$-norm

We validate the convergence property of the reweighted method proposed in Lemma 1 when solving $\ell_{2,1}$-norm with symmetric constraint. Given different sizes of input metrics, we test the change of the objective function in Eq. (4). Two sizes of metrics are shown in Fig. 2, the first (a) is a $20 \times 20$ metric and the second (b) is of size 200.

It can be found that: 1. the reweighted method converges at last; 2. this method converges very quickly and the time to converge may depend on the size of the input. The larger the metric, the more iterations it needs to converge. When the dimension is low, it can converge in about 5 iterations. While with a larger input of size 200, the objective value converges in about 10 iterations. In addition, on the larger size input, time comparison is conducted between the reweighted method and the symmetric iterative projection one in [8]. We force the two methods to achieve nearly the same objective value in Eq. (4) at last and the time comparison is conducted 100 times in total. The average convergence time (in second) for reweighted one and symmetric projection one are 0.0093 (0.0012) and 0.0665 (0.0023) respectively. Values in brackets are standard deviations of time. This result reveals our reweighted solver can be more efficient when dealing with $\ell_{2,1}$-norm regularization on metrics, since at each step it takes norm on row and column into consideration simultaneously and thus is symmetric projection free.

References


