Applied Intelligence
https://doi.org/10.1007/s10489-022-03419-1

Low-rank robust online distance/similarity learning based on the rescaled hinge loss

Davood Zabihzadeh[1] · Amar Tuama[2] · Ali Karami-Mollaee[3] · Seyed Jalaleddin Mousavirad[1]

Accepted: 21 February 2022

Abstract
An important challenge in metric learning is scalability to both size and dimension of input data. Online metric learning algorithms are proposed to address this challenge. Existing methods are commonly based on Passive/Aggressive (PA) approach. Hence, they can rapidly process large volumes of data with an adaptive learning rate. However, these algorithms are based on the Hinge loss and so are not robust against outliers and label noise. We address the challenges by formulating the online Distance/Similarity learning problem with the robust Rescaled Hinge loss function. The proposed model is rather general and can be applied to any PA-based online Distance/Similarity algorithm. To achieve scalability to data dimension, we propose low-rank online Distance/Similarity methods that learn a rectangular projection matrix instead of a full Mahalanobis matrix. The low-rank approaches not only reduce the computational cost but also keep the discrimination power of the learned metrics. Also, current online methods usually assume training triplets or pairwise constraints exist in advance. However, this assumption does not hold, and generating triplets using available batch sampling methods is both time and space consuming. We address this issue by developing an efficient, yet effective robust one-pass triplet construction algorithm. We conduct several experiments on datasets from various applications. The results confirm that the proposed methods significantly outperform state-of-the-art online metric learning methods in the presence of label noise and outliers by a large margin.

Keywords Metric learning · Rescaled hinge loss · Robust algorithm · Label noise · Online distance/similarity learning · One pass triplet construction

1 Introduction

The performance of many machine learning and data mining algorithms depends on the metric used to compute the distance between data. Generic measures such as Euclidean or Cosine similarity in an input space often fail to discriminate different classes or clusters of data. Therefore, learning an optimal Distance/Similarity function from training information is actively studied in the last decade.

Distance Metric Learning (DML) methods aim to bring semantically similar data items together while keeping dissimilar ones at a distance. One major challenge for DML algorithms is scalability to both the size and dimension of input data [1]. For processing massive volumes of data generated in today's applications, online methods are proposed.

Many of these algorithms are based on the PA [2–8]. The main advantages of PA-based methods are 1) closed-form solution and 2) adaptive learning rate leading to a high convergence rate. However, the following challenges are still needed to be addressed:

1- These algorithms are based on the Hinge loss and hence are not robust against outliers and label noise data. Nowadays many modern datasets are collected from the Internet using crowdsourcing or similar techniques. Thus, examples with wrong labels are usual in these datasets that can considerably degrade the performance of existing online DML methods.
2- Most DML algorithms learn a metric from pair or triplet side information defined as:

✉ Davood Zabihzadeh
d.zabihzadeh@hsu.ac.ir

[1] Department of Computer Engineering, Hakim Sabzevari University, Sabzevar, Iran

[2] Computer Engineering Department, Engineering Faculty, Ferdowsi University of Mashhad, Mashhad, Iran

[3] Electrical and Computer Engineering Faculty, Hakim Sabzevari University, Sabzevar, Iran

Published online: 20 April 2022

$$S = \{(\boldsymbol{x}_i, \boldsymbol{x}_i) | \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are similar}\}$$
$$D = \{(\boldsymbol{x}_i, \boldsymbol{x}_i) | \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are dissimilar}\}$$
$$T = \{(\boldsymbol{x}_i, \boldsymbol{x}_i^+, \boldsymbol{x}_i^-) | \boldsymbol{x}_i \text{ should be more closer to } \boldsymbol{x}_i^+ \text{ than to } \boldsymbol{x}_i^-\}$$

Existing online methods [3–8] usually assume training triplets or pairs exist in advance. However, this assumption does not hold, and generating constraints by available batch sampling methods is both time and space consuming. Thus, we need an efficient one-pass sampling algorithm for online tasks.

3- Another important challenge in online DML applications, particularly in machine vision domain, is the *high dimension* of input data. Many existing methods learn Mahalanobis distance [3, 5, 6, 8] or bilinear similarity [2, 3] that require $O(d^2)$ parameters (d indicates the data dimension). Therefore, these methods are infeasible in high dimensional environments.

The main contributions of the paper to overcome these issues are as follows:

1- We address the first challenge by formulating the online Distance/Similarity learning task using the *robust Rescaled Hinge loss* [9]. The proposed model is rather *general*, and we can easily apply it to any existing PA-based methods. It significantly improves the robustness of existing methods in the presence of label noise without increasing their computational complexity.
2- The second challenge is tackled by developing an efficient robust *one-pass triplet construction* algorithm in our work.
3- Finally, we overcome the third challenge by developing the *low-rank* versions of the proposed methods that learn a rectangular projection matrix instead of a full Mahalanobis matrix. These approaches not only decrease the computational cost significantly but also retain the predictive performance of the learned metrics. Also, we can easily replace the low-rank projection matrix with a nonlinear deep neural network model. Therefore, extending our methods for online deep metric learning is straightforward.

Table 1 summarizes the main notations used throughout the paper. The rest of the paper is organized as follows: Section 2 reviews related work. In Section 3, we present the formulation of the online Distance/Similarity learning problem using the Rescaled Hinge loss as well as the development of the proposed algorithms. Experiments conducted to evaluate the proposed methods are discussed in Section 4. Finally, Section 5 concludes with remarks and recommendations for future work.

2 Related work

DML is a well-studied problem and attracts a lot of interest in the last decade. We refer interested readers to the surveys [1, 10] for a complete review of existing work. In this section, we only focus on related online Distance/Similarity learning algorithms. Most existing online learning methods learn a Mahalanobis distance [4–8] or a bilinear similarity [2, 3]. However, some more generic measures such as [5, 11] are also presented.

Mahalanobis-based methods learn a matrix $\boldsymbol{M} \succcurlyeq \boldsymbol{0}$ given by:

$$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{1}$$

Since the matrix $\boldsymbol{M} \succcurlyeq \boldsymbol{0}$, it can be decomposed as $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^T$ where $\boldsymbol{L} \in \mathbb{R}^{d \times r}$ and $r = rank(\boldsymbol{M})$. Therefore, Mahalanobis distance learning is equivalent to find a linear transformation $\boldsymbol{L}$ in the input space. Instead, bilinear similarity-based methods learn a similarity matrix $\boldsymbol{M}$ given by:

$$S_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = \boldsymbol{x}_i^\top \boldsymbol{M} \boldsymbol{x}_j \tag{2}$$

The optimization problem of both Mahalanobis and bilinear methods is formulated based on the PA approach as follows:

$$\boldsymbol{M}_{t+1} = \arg\min_{\boldsymbol{M}} reg(\boldsymbol{M}, \boldsymbol{M}_t) + C\xi$$
$$\text{subject to} \quad l(\boldsymbol{M}, R_t) \le \xi, \qquad \xi \ge 0, \ldots\ldots\ldots \boldsymbol{M} \succcurlyeq 0$$

where $\boldsymbol{M}_t$ is the current Distance/Similarity matrix at time step t, $reg(\boldsymbol{M}, \boldsymbol{M}_t)$ is a regularization term, and $l(\boldsymbol{M}, R_t)$ indicates the margin-based Hinge loss function. In distance-based methods, the Hinge loss is defined as:

$$l(\boldsymbol{M}, (\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)) = \max\left\{0, 1 + d_{\boldsymbol{M}}(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2 - d_{\boldsymbol{M}}(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2\right\} \tag{4}$$

whereas it is defined in similarity-based methods as:

$$l(\boldsymbol{M}, (\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)) = \max\left\{0, 1 - S_M(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2 + S_M(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2\right\} \tag{5}$$

OASIS[1] [2] is a popular bilinear similarity learning method that uses the Frobenius norm as a regularization term, i.e. $reg(\boldsymbol{M}, \boldsymbol{M}_t) = \frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}_t\|_F^2$. OASIS eliminates the p.s.d

[1] Online Algorithm for Scalable Image Similarity

(positive semi-definite) constraint for scalability reasons. However, this property is extremely useful to produce a low-rank metric as well as to prevent overfitting.

OKS[2] [3] extends OASIS in the feature space of an RKHS kernel. Also, [3] presents OMKS which is the extension of OKS for multi-modal data.

ODML[3] and OMDML[4] [4] are similar to OKS and OMKS respectively, but instead of bilinear similarity, they learn Mahalanobis distance. To enforce the p.s.d constraint, these methods use full Eigen value decomposition that involves $O(d^3)$ operations at each stage. Therefore, they are infeasible for high-dimensional DML tasks. To address this problem, LSMDML[5] [8] utilizes DRP (Dual Random Projection) [12] in an online multi-modal environment to enforce the p.s.d constraint.

SLMOML[6] [5] is the online version of the seminal ITML[7] [13] method. It uses the *logdet* regularization term that automatically enforces the p.s.d constraint at each time step. However, it has a low convergence rate and still requires $O(d^2)$ parameters.

In [6] a large-scale local online Distance/Similarity framework is presented. It learns multiple metrics for the task at hand, one metric per class in the dataset. Each metric in this framework consists of a global and a local component learned simultaneously. Having a common component for local metrics shares discriminating information among them and efficiently reduces the overfitting problem.

OPML[8] [7] is an online DML method that learns projection matrix $\boldsymbol{L}$ (see eq. (1)) directly, so it does not require imposing the p.s.d constraint. In practice, $\boldsymbol{L}$ has a rectangular form ($\boldsymbol{L} \in \mathbb{R}^{d \times r}$, $r \ll d$). However, OPML learns a square $d \times d$ matrix and obtains a closed-form solution with the $O(d^2)$ time complexity. Also, it adopts the Frobenius norm regularization term and the popular Hinge loss function. An interesting feature of OPML is its triplet sampling strategy which constructs triplets from incoming data in an online setting.

OAHU[9] [14] aims to dynamically adapt the complexity of the model and to effectively utilize the input constraints during the learning process. For this purpose, this method introduces the Adaptive-Bound Triplet Loss (ABTL) instead of the commonly used Hinge loss. Also, it uses an over-complete neural network model and connects a different MEI (Metric Embedding Layer) to each hidden layer of the network. The overall loss is considered as a weighted average loss of each MEI.

[2] Online Kernel Similarity Learning
[3] Online Distance Metric Learning
[4] Online Multi-Modal Distance Metric Learning
[5] Large-Scale Multi-modal Distance Metric Learning
[6] Scalable Large Margin Online Metric Learning
[7] Information-Theoretic Metric Learning
[8] One-Pass Metric Learning
[9] Online metric learning with Adaptive Hedge Update

Table 2 summarizes the advantages and limitations of existing online metric learning methods. As seen, all studied online Distance/Similarity models are *non-roust against label noise*. These methods assume that the input training information is perfect. However, this assumption may be wrong in practical machine vision applications where this information is collected from the Internet by crowdsourcing or similar techniques. Although some robust DML methods such as [7, 8, 15, 16] are presented, these methods are focused on batch settings. Among them, only Bayesian approaches [7, 16] can be extended for online settings. However, although Bayesian learning helps to avoid over-fitting in a small or a dataset with noisy features, it is less effective to deal with the more complicated problem (i.e., label noise).

Many metric learning algorithms such as [8, 11, 17, 18] generate triplets from training data using the following *batch* procedure. Each data point $\boldsymbol{x}_i$ is considered *similar* to its k nearest neighbors with the same label (named *target neighbors* of $\boldsymbol{x}_i$). Suppose $\boldsymbol{x}_j$ is a *target neighbor* of $\boldsymbol{x}_i$. The *imposters* of $\boldsymbol{x}_i$ are any data point $\boldsymbol{x}_l$ of a different class (i.e., $y_i \neq y_l$) which *violates* the following condition:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) + margin < d(\boldsymbol{x}_i, \boldsymbol{x}_l)$$

where d is a distance measure such as Euclidean.

The data point $\boldsymbol{x}_i$ is set dissimilar to any of its imposters. Then, the triplets are formed by the natural join of similar and dissimilar pairs. Figure 1 illustrates the concepts of *target neighbors* and *imposters*.

Generating triplets using this procedure is both time and space consuming and is not feasible for online tasks. Although an online triplet construction algorithm presented in OPML that is very efficient in terms of computational cost, it does not consider the distribution and structure of data. Therefore, it has a lower performance in comparison with that of the batch algorithm.

3 Proposed methods

In this section, we first derive a general form of objective functions in existing online Similarity/Distance methods. Then, we propose its robust variant based on the Rescaled Hinge loss. After, we develop some algorithms that *efficiently* solve the problem based on HQ.

3.1 General Form of Objective Functions in Online Similarity/Distance methods:

As observed, many Distance/Similarity algorithms are based on the margin-based Hinge loss function (l_{hinge}). Let define the variable z_t as follows:

Table 1 Summary of the main notations

Notation	Description
$\boldsymbol{M}_t$	Distance or similarity matrix at time t, $\boldsymbol{M}_t \in \mathbb{R}^{d \times d}$
$\boldsymbol{L}_t$	Linear projection matrix at time t. $\boldsymbol{M}_t = \boldsymbol{L}_t \boldsymbol{L}_t^T$, $\boldsymbol{L}_t \in \mathbb{R}^{d \times r}$, $r \ll d$
$R_t = (\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)$	Incoming triplet at time t. $\boldsymbol{p}_t$: anchor, $\boldsymbol{p}_t^+$: positive, $\boldsymbol{p}_t^-$:negative
$S_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$	Bilinear similarity function
$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$	Mahalanobis distance
$\boldsymbol{M} \succcurlyeq 0$	p.s.d matrix $\boldsymbol{M}$
l_{hinge}	The Hinge loss. $l_{hinge}(z_t) = \max\{0, 1 - z_t\}$
l_{rhinge}	The Rescaled Hinge loss
η	Rescaled parameter in l_{rhinge}
HQ	Half-Quadratic
v_t	Auxiliary variable of the HQ algorithm at time t
$MaxHQIter$	Maximum number of iterations in the HQ algorithm
ξ	Slack variable
C_t	Weight of the triplet at time t
τ	Adaptive learning rate

$$z_t = \begin{cases} S_M(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2 - S_M(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2, & \textit{For similarity-based methods} \quad (6) \\ d_{\boldsymbol{M}}(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2 - d_{\boldsymbol{M}}(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2, & \textit{Mahalanobis-based methods} \quad (7) \end{cases}$$

The Hinge loss is then can be written as:

$$l\left(\boldsymbol{M}, (\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)\right) = \max\{0, 1 - z_t\} \quad (8)$$

Figure 2 shows the loss function. As seen, the loss linearly grows for $z \leq 1$ with no bound. The unboundedness of the Hinge loss function causes the noisy labeled data and outliers to have a large effect on the training process that results in poor performance for the learned Distance/Similarity measure.

Most existing Distance/Similarity learning methods can be formulated as follows:

$$\boldsymbol{M}_{t+1} = \arg\min_{\boldsymbol{M}} \operatorname{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C l_{hinge}(z_t) \quad [\text{subject to } \boldsymbol{M} \succcurlyeq 0] \quad (9)$$

Note that the constraint $\boldsymbol{M} \succcurlyeq 0$ is not adopted in all methods, so we enclose it by bracket. We can derive many existing methods from this generic optimization problem. For example, if we consider $\operatorname{reg}(\boldsymbol{M}, \boldsymbol{M}_t) = \frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}_t\|_F^2$ and omit the $\boldsymbol{M} \succcurlyeq 0$ constraint, then by defining z_t according to (6), we obtain the OASIS [2] and OKS[10] [3] optimization problems. Also, if we consider z_t equal to (7), the optimization problem (9) reduces to the OPML [7]. Similarly, by setting $\operatorname{reg}(\boldsymbol{M}, \boldsymbol{M}_t) = \frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}_t\|_F^2$, z_t equal to (7), and enforcing $\boldsymbol{M} \succcurlyeq 0$, we reach the optimization problem in [4]. Finally, if we set $\operatorname{reg}(\boldsymbol{M}, \boldsymbol{M}_t) = D_{ld}(\boldsymbol{M}, \boldsymbol{M}_t) = \operatorname{trace}\left(\boldsymbol{M}\boldsymbol{M}_t^{-1}\right) - \operatorname{logdet}\left(\boldsymbol{M}\boldsymbol{M}_0^{-1}\right) - d$ and drop the $\boldsymbol{M} \succcurlyeq 0$ constraint, we obtain the optimization problem of [5].

One approach to alleviate the effect of label noise data in PA-based problems (such as eq. (9)) is to select a small value for the hyper-parameter C. However, it causes lower values for the adaptive learning rate. Instead, we propose to replace the Hinge function with the robust Rescaled Hinge loss.

3.2 Robust variant of the General Objective Function:

The Rescaled Hinge loss is defined as:

$$l_{rhinge}(z) = \beta\left[1 - \exp\left(-\eta l_{hinge}(z)\right)\right] \quad (10)$$

Figure 3 shows the diagram of the $l_{rhinge}(z)$ loss function with different values of η. In this function, η is a rescaling parameter and $\beta = 1/(1 - \exp(-\eta))$ is just a normalizing constant that ensures $l_{rhinge}(0) = 1$. As seen, this loss function is more robust than the Hinge against the outliers and data contaminated with label noise. We can adjust the degree of robustness using the η parameter. Also, the Hinge loss can be regarded as a special case of the Rescaled Hinge. More specifically, $l_{rhinge}(z)$ becomes $l_{hinge}(z)$ as $\eta \to 0$.

By replacing the Hinge loss function with the Rescaled Hinge loss in eq. (9), we obtain the following optimization problem for online robust Distance/Similarity learning.

$$\boldsymbol{M}_{t+1} = \arg\min_{\boldsymbol{M}} \operatorname{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C l_{\boldsymbol{rhinge}}(z_t) \quad [\text{subject to } \boldsymbol{M} \succcurlyeq 0] \quad (11)$$

[10] Online Kernel Similarity

Table 2 Advantages and limitations of existing online metric learning methods

Method	Description	Advantages	Limitations
OASIS [2]	Using bilinear similarity measure. Omitting the p.s.d constraint. Adopting the Hinge Loss.	Scalable	Non-Robust More prone to overfitting
OKS[a] [3]	Extending OASIS in the feature space of a kernel.	Learning non-linear projection in input space	Non-Robust Not providing an online triplet sampling algorithm. More prone to overfitting. The model size increases over time.
OMKS [3]	Extending OKS for multiple kernel learning. It combines the result of each kernel using the Hedge algorithm.	Learning non-linear projection in the input space. High flexibility of the learned similarity.	Non-Robust. Not providing an online triplet sampling algorithm. More prone to overfitting. The model size increases over time.
ODML[b] [4]	Similar to OASIS but uses Mahalanobis distance. Considers the p.s.d constraint.	Non-Scalable. Resistance against overfitting.	Non-Robust. Non-Scalable. Not providing an online triplet sampling algorithm.
OMDML [4]	Extends ODML for multiple kernel learning. Combines the result of each kernel using the Hedge algorithm	Learns a non-linear projection in the input space. High flexibility of the learned metric. Resistance against overfitting.	Non-Robust. Non-Scalable. Not providing an online triplet sampling algorithm.
SLMOML [5]	Online version of the seminal ITML method. Uses logdet regularization.	Scalable to some extent. Still, it has $O(d^2)$ parameters. Resistance against overfitting.	Non-Robust. Not providing an online triplet sampling algorithm.
LPA-ODML[c] [6]	Learns multiple metrics. Each metric consists of a global (shared) and a local component. Utilizes DRP to achieve scalability.	Learns non-linear projection. Good discrimination power. Resistance against overfitting.	Non-Robust. Not providing an online triplet sampling algorithm.
OPML [7]	Learns projection matrix directly. Provides a one-pass triplet construction algorithm	Scalable to some extent. Still, needing $O(d^2)$ parameters.	Non-Robust. The triplet sampling algorithm does not consider the structure of data.
OAHU [14]	An online deep metric learning method. Learns a metric per layer in the network. Combines the metrics using the Hedge algorithm.	An end-to-end metric learning. Good discrimination power. Dynamically adapts the complexity of the model.	Non-Robust. Non-Convex. Not providing an online triplet sampling algorithm. Too many parameters. Needs a metric embedding per layer.
LSMDML[8]	Learns a metric for each source of multi-modal data. Fuses the metrics using a PA-based method.	Good discrimination power. Resistance against overfitting.	Non-Robust. Not providing an online triplet sampling algorithm.

[a] Online Kernel Similarity Learning

[b] Online Distance Metric Learning

[c] Local Passive/Aggressive Online Distance Metric Learning

In the next subsections, we derive two efficient algorithms that efficiently solve the above optimization problem in online fashion.

3.3 The proposed Robust methods

Since the Rescaled Hinge loss is not convex, we need an efficient algorithm to solve the optimization problem (11). The proposed algorithms are based on HQ (Half Quadratic) which is an efficient alternating approach for optimizing non-convex problems. The main idea of HQ is to add an auxiliary variable such as v to the problem using Conjugate theory [19], such that the new optimization problem becomes quadratic to the main variable (with the same optimal solution as the original non-convex problem).

Since $l_{rhinge}(z) = \beta[1 - \exp(-\eta l_{hinge}(z))]$, we can obtain the following problem which is equivalent to (11).

$$\boldsymbol{M}_{t+1} = \arg\max_{\boldsymbol{M}} -\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\exp\left(-\eta l_{hinge}(z_t)\right) \quad [\text{subject to } \boldsymbol{M} \succcurlyeq 0] \tag{12}$$

According to the definition of conjugate function, we have (refer to the Appendix A of [9]),

$$\exp\left(-\eta l_{hinge}(z)\right) = \sup_{v<0}\left(\eta l_{hinge}(z)v - g(v)\right) \tag{13}$$

where $g(v) = -v\log(-v) + v, \quad (v < 0)$. By substituting eq. (13) in (12), we obtain

$$\begin{aligned} f(\mathbf{M}) &= -\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\exp\left(-\eta l_{hinge}(z_t)\right) \\ &= -\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\sup_{v_t<0}\left(\eta l_{hinge}(z_t)v_t - g(v_t)\right) \\ &= \sup_{v_t<0}\left(-\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\left(\eta l_{hinge}(z_t)v_t - g(v_t)\right)\right) \end{aligned} \tag{14}$$

The third relation in (14) holds since $-\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t)$ is constant regarding v. Using (14), we can rewrite (12) as:

$$\left(\boldsymbol{M}_{t+1}, \boldsymbol{v}_t^*\right) = \arg\max_{\boldsymbol{M}, v_t} -\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\left(\eta l_{hinge}(z_t)v_t - g(v_t)\right) \quad [\text{subject to } \boldsymbol{M} \succcurlyeq 0] \tag{15}$$

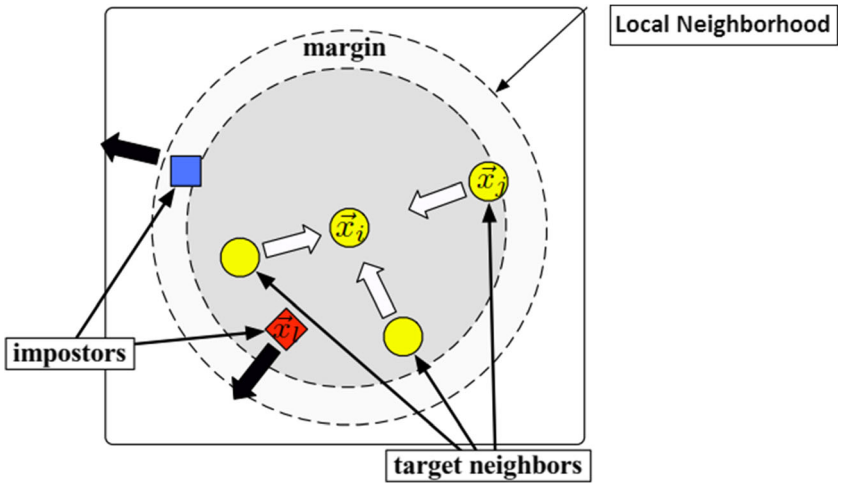

Fig. 1 Illustration of target neighbors and imposters of $\boldsymbol{x}_i$ [17]

To solve the above problem, we use the alternating optimization approach. First, given $\boldsymbol{M}$, we optimize (13) over $\boldsymbol{v}_t$ and then given $\boldsymbol{v}_t$, we optimize it over $\boldsymbol{M}$. Suppose $\boldsymbol{M}^{(s)}$ is given (the superscript s indicates the iteration number), then (15) is equivalent to:

$$v_t^{(s)} = \arg\max_{v_t} \quad \left(\eta l_{hinge}(z_t) v_t - g(v_t)\right) \tag{16}$$

The above equation has a closed-form solution obtained by setting its derivative with respect to $\boldsymbol{v}_t$ equal to zero.

$$v_t^{(s)} = -\exp\left(-\eta l_{hinge}(z_t)\right) \tag{17}$$

After obtaining $v_t^{(s)}$, we optimize the eq. (15) respecting to $\boldsymbol{M}$ as follows:

$$\boldsymbol{M} = \arg\max_{\boldsymbol{M}} - \mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\eta v_t \, l_{hinge}(z_t) \quad [\text{subject to } \boldsymbol{M} \succcurlyeq 0] \tag{18}$$

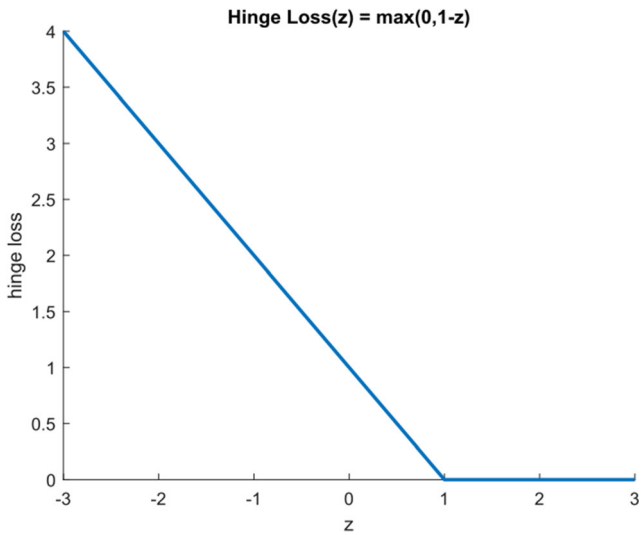

Fig. 2 The margin-based Hinge loss function. The loss linearly grows for $z \leq 1$ with no bound

The above problem is equivalent to:

$$\boldsymbol{M} = \arg\min_{\boldsymbol{M}} \quad \mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C_t l_{hinge}(z_t) \quad \text{subject to } [\boldsymbol{M} \succcurlyeq 0], \quad l_{hinge}(z_t) \leq \xi, \quad \xi \geq 0 \tag{19}$$

where $C_t = C\beta\eta(-v_t)$. The robustness of the optimization problem (19) can be explained using the penalty factor C_t. Suppose the current triplet R_t contains noisy labeled data, so the hinge function ($l_{hinge}(z_t)$) returns a large loss for R_t. Thus, $C_t = C\beta\eta(-v_t) = C\beta\eta \exp(-\eta l_{hinge}(z_t))$ approaches zero. Therefore, R_t has less effect on the learning process.

The obtained optimization problem, unlike existing models, assigns an adaptive weight (C_t) for each incoming triplet. By adjusting reg($\boldsymbol{M}$, $\boldsymbol{M}_t$), p.s.d constraint, and z_t, we can obtain a family of robust Distance/Similarity learning methods. For instance, we develop two proposed algorithms named Robust_OASIS and Robust_ODML.[11] These algorithms can be considered as robust variants of existing OASIS [2] and ODML [4] respectively.

3.3.1 Robust_OASIS

The robust similarity-based algorithm can be derived from the general optimization problem (15) by the following settings:

$\mathrm{reg}(\boldsymbol{M}, \boldsymbol{M}_t) = \frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}_t\|_F^2$, drop $\boldsymbol{M} \succcurlyeq 0$ constraint, and define z_t according to (6).

Then, the following optimization problem is achieved:

$$(\boldsymbol{M}_{t+1}, \boldsymbol{v}_t^*) = \arg\max_{\boldsymbol{M}, v_t} -\frac{1}{2}\|\boldsymbol{M} - \boldsymbol{M}_t\|_F^2 + C\beta\left(\eta l_{hinge}(z_t) v_t - g(v_t)\right) \tag{20}$$

The solution of the above problem is obtained by iteratively computing v_t from eq. (17) and then optimizing $\boldsymbol{M}$ by solving the following optimization problem.

[11] Robust Online Distance Metric Learning

$$\boldsymbol{M} = \arg\min_{\boldsymbol{M}} \ \frac{1}{2}\|\boldsymbol{M}-\boldsymbol{M}_t\|_F^2 + C_t\xi$$
$$\text{subject to } l(p_t,p_t^+,p_t^-) = 1-S_{\boldsymbol{M}}(\boldsymbol{p}_t,\boldsymbol{p}_t^+) + \boldsymbol{S}_{\boldsymbol{M}}(\boldsymbol{p}_t,\boldsymbol{p}_t^-) \le \xi, \qquad \xi \ge 0 \tag{21}$$

The problem (21) has a similar solution to that obtained in [2].

$$\boldsymbol{M}_{t+1} = \boldsymbol{M}_t + \tau \boldsymbol{A}_t$$
$$\text{where } \tau = \min\left(C_t, \frac{l(\boldsymbol{p}_t,\boldsymbol{p}_t^+,\boldsymbol{p}_t^-)}{\|\boldsymbol{A}_t\|_F^2}\right) \text{ and } \boldsymbol{A}_t = \boldsymbol{p}_t(\boldsymbol{p}_t^+-\boldsymbol{p}_t^-)^\top \tag{22}$$

The main difference is that now the learning rate τ is bounded to the adaptive triplet weight C_t instead of the constant C in the OASIS. Algorithm 1 summarizes the steps of Robust-OASIS.

$\text{reg}(\boldsymbol{M},\boldsymbol{M}_t) = \frac{1}{2}\|\boldsymbol{M}-\boldsymbol{M}_t\|_F^2$, enforce $\boldsymbol{M} \succcurlyeq 0$ constraint, and define z_t according to (7).

We then obtain the following optimization problem:

$$(\boldsymbol{M}_{t+1}, \boldsymbol{v}_t^*) = \arg\max_{\boldsymbol{M},v_t} \ -\frac{1}{2}\|\boldsymbol{M}-\boldsymbol{M}_t\|_F^2 + C\beta(\eta l_{hinge}(z_t)v_t - g(v_t)) \tag{23}$$
$$\text{subject to } \boldsymbol{M} \succcurlyeq 0$$

In a similar way to the Robust-OASIS, we obtain the solution by iteratively computing v_t from the eq. (17) and then optimizing $\boldsymbol{M}$ by solving the following optimization problem.

$$\boldsymbol{M} = \arg\min_{\boldsymbol{M}} \ \frac{1}{2}\|\boldsymbol{M}-\boldsymbol{M}_t\|_F^2 + C_t\xi$$
$$\text{subject to } l(\boldsymbol{p}_t,\boldsymbol{p}_t^+,\boldsymbol{p}_t^-) = 1 + d_{\boldsymbol{M}}^2(\boldsymbol{p}_t,\boldsymbol{p}_t^+) - d_{\boldsymbol{M}}^2(\boldsymbol{p}_t,\boldsymbol{p}_t^-) \le \xi, \quad \xi \ge 0, \quad \boldsymbol{M} \succcurlyeq 0 \tag{24}$$

The solution of the above problem is similar to that of [4].

Algorithm1. Robust-OASIS

Inputs: $C, \eta, MaxHQIter$

Output: Similarity matrix $\boldsymbol{M}$

1. Initialize $\boldsymbol{M}$ with the Identity matrix.
2. Set $v = 1$
3. for $t = 1,2,\ldots,T$
   - 3.1. Receive an input triplet $(\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)$
   - 3.2. Compute $\boldsymbol{A}_t = \boldsymbol{p}_t(\boldsymbol{p}_t^+ - \boldsymbol{p}_t^-)^\top$, and $\|\boldsymbol{A}_t\|_F^2$
   - 3.3. for $s = 1,2,\ldots,MaxHQIter$
     - 3.3.1. Update v from (17)
     - 3.3.2 Compute the triplet weight $C_t = C\beta\eta(-v)$.
     - 3.3.3. Obtain the learning rate $\tau = \min(C_t, \frac{l(\boldsymbol{p}_t,\boldsymbol{p}_t^+,\boldsymbol{p}_t^-)}{\|\boldsymbol{A}_t\|_F^2})$
     - 3.3.4. Update $\boldsymbol{M}$ from (22)

   end;

3.3.2 Robust_ODML

The robust Mahalanobis distance learning algorithm can be derived from the general optimization problem (15) by the following settings:

$$\boldsymbol{M}_{t+1} = \boldsymbol{M}_t + \tau \boldsymbol{A}_t$$
$$\text{where } \tau = \min\left(C_t, \frac{l(\boldsymbol{p}_t,\boldsymbol{p}_t^+,\boldsymbol{p}_t^-)}{\|\boldsymbol{A}_t\|_F^2}\right) \text{ and} \tag{25}$$
$$\boldsymbol{A}_t = (\boldsymbol{p}_t-\boldsymbol{p}_t^-)(\boldsymbol{p}_t-\boldsymbol{p}_t^-)^\top - (\boldsymbol{p}_t-\boldsymbol{p}_t^+)(\boldsymbol{p}_t-\boldsymbol{p}_t^+)^\top$$

Algorithm 2 summarizes the steps of Robust-ODML.

Algorithm2. Robust-ODML

Inputs: $C, \eta, MaxHQIter$

Output: Distance matrix $\boldsymbol{M}$

1. Initialize $\boldsymbol{M}$ with the Identity matrix.
2. Set $v = 1$
3. for $t = 1,2,\ldots,T$
   - 3.1. Receive an input triplet $(\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)$
   - 3.2. Compute $\boldsymbol{A}_t = (\boldsymbol{p}_t - \boldsymbol{p}_t^-)(\boldsymbol{p}_t - \boldsymbol{p}_t^-)^\top - (\boldsymbol{p}_t - \boldsymbol{p}_t^+)(\boldsymbol{p}_t - \boldsymbol{p}_t^+)^\top$, and $\|\boldsymbol{A}_t\|_F^2$
   - 3.3. for $s = 1,2,\ldots,MaxHQIter$
     - 3.3.1. Update v from (17)
     - 3.3.2 Compute the triplet weight $C_t = C\beta\eta(-v)$.
     - 3.3.3. Obtain the learning rate $\tau = \min(C_t, \frac{l(\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)}{\|\boldsymbol{A}_t\|_F^2})$
     - 3.3.4. Update $\boldsymbol{M}$ from (25)
4. Enforce the p.s.d constraint: $\boldsymbol{M} = psd(M)$

end;

To enforce the p.s.d constraint, the naive approach is to perform the full Eigen value decomposition of matrix $\boldsymbol{M}$ and then set its negative Eigen values to zero. This approach requires $O(d^3)$ operations, so it is infeasible for high-dimensional DML tasks. Although some improved methods are available [6, 8, 12], we address this problem by developing the low-rank versions of the proposed algorithms in the following subsections.

3.4 Low-rank Robust Distance/Similarity learning methods

Instead of a full Mahalanobis matrix $\boldsymbol{M} \in \mathbb{R}^{d \times d}$, the proposed low-rank methods learn a rectangular projection matrix $\boldsymbol{L} \in \mathbb{R}^{d \times r}$ where $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$and r is the rank of $\boldsymbol{M}$. We follow this approach to achieve the low-rank variants of the proposed method since: 1) it automatically enforces the p.s.d constraint, 2) in many real applications data lie on a latent subspace with dimensionality $r \ll d$. Thus, this approach requires fewer parameters. An important problem is how to adjust the hyper parameter r. While some sophisticated methods like Bayesian variational inference [16] or low-rank approximation [20] exist that can automatically adjust the value of r; here, we simply use the cross-validation.

The optimization problem for low-rank online Distance/Similarity learning is formulated as:

$$\boldsymbol{L}_{t+1} = \arg\max_{\boldsymbol{L}} -\mathrm{reg}(\boldsymbol{L}, \boldsymbol{L}_t) + C\beta\big(\eta l_{hinge}(z_t)v_t - g(v_t)\big) \quad (26)$$

If we rewrite both the bilinear similarity and the Mahalanobis distance as functions of $\boldsymbol{L}$ as follows:

$$S_{\boldsymbol{L}}(\boldsymbol{p}, \boldsymbol{q})^2 = \boldsymbol{p}^\top \boldsymbol{M}\boldsymbol{q} = \boldsymbol{p}^\top \boldsymbol{L}\boldsymbol{L}^\top \boldsymbol{q} = \big(\boldsymbol{L}^T\boldsymbol{p}\big)^\top (\boldsymbol{L}^\top \boldsymbol{q}) \quad (27)$$

$$d_{\boldsymbol{L}}(\boldsymbol{p}, \boldsymbol{q})^2 = (\boldsymbol{p}-\boldsymbol{q})^\top \boldsymbol{M}(\boldsymbol{p}-\boldsymbol{q}) = (\boldsymbol{p}-\boldsymbol{q})^\top \boldsymbol{L}\boldsymbol{L}^\top(\boldsymbol{p}-\boldsymbol{q}) = \|\boldsymbol{L}^\top \boldsymbol{p} - \boldsymbol{L}^\top \boldsymbol{q}\|_2^2, \quad (28)$$

Then, the bilinear similarity learning is equivalent to finding a linear projection $\boldsymbol{L}$ and then applying dot product to the inputs in the projected space. Similarly, Mahalanobis distance learning corresponds to compute the Euclidean distance after transforming the inputs by $\boldsymbol{L}$.

The z_t variable can be expressed in terms of $S_{\boldsymbol{L}}$ and $d_{\boldsymbol{L}}$ as:

$$z_t = \begin{cases} S_{\boldsymbol{L}}(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2 - S_{\boldsymbol{L}}(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2, & \textit{For similarity-based methods} \quad (29) \\ d_{\boldsymbol{L}}(\boldsymbol{p}_t, \boldsymbol{p}_t^-)^2 - d_{\boldsymbol{L}}(\boldsymbol{p}_t, \boldsymbol{p}_t^+)^2, & \textit{Mahalanobis-based methods} \quad (30) \end{cases}$$

Now, we can easily derive the proposed low-rank robust similarity learning algorithm named Robust-LOSL[12] from the generic optimization problem (26) with the following settings: $\mathrm{reg}(\boldsymbol{L}, \boldsymbol{L}_t) = \frac{1}{2}\|\boldsymbol{L} - \boldsymbol{L}_t\|_2^2$, define z_t according to (29).

The obtained optimization problem can be solved by iteratively computing v_t from the eq. (17) and then optimizing $\boldsymbol{L}$ by

[12] Robust Low-rank Online Similarity Learning

solving the following optimization problem:

$$\boldsymbol{L}_{t+1} = \arg\min_{\boldsymbol{L}} \ \frac{1}{2}\|\boldsymbol{L}-\boldsymbol{L}_t\|_F^2 + C_t l_{hinge}(z_t) \tag{31}$$

The above *unconstrained* optimization problem is non-convex. However, we can solve it efficiently by optimizing a simple linear neural network model parameterized by $\boldsymbol{L}$ as illustrated in Fig. 4.

The sub-gradient of the loss function with respect to $\boldsymbol{L}$ can be computed from the following equation:

$$\begin{aligned}\frac{\partial l_t}{\partial \boldsymbol{L}} &= (\boldsymbol{L}-\boldsymbol{L}_t) + C_t\left[\boldsymbol{p}_t\boldsymbol{p}_t^{-\top} + \boldsymbol{p}_t^{-}\boldsymbol{p}_t^{\top} - \boldsymbol{p}_t\boldsymbol{p}_t^{+\top} - \boldsymbol{p}_t^{+}\boldsymbol{p}_t^{\top}\right]\boldsymbol{L} \\ &= (\boldsymbol{L}-\boldsymbol{L}_t) - C_t\left[\boldsymbol{p}_t\left(\boldsymbol{p}_t^{+}-\boldsymbol{p}_t^{-}\right)^{\top} + \left(\boldsymbol{p}_t^{+}-\boldsymbol{p}_t^{-}\right)\boldsymbol{p}_t^{\top}\right]\boldsymbol{L} \\ &\quad (\boldsymbol{L}-\boldsymbol{L}_t) - C_t\left[\boldsymbol{A}_t + \boldsymbol{A}_t^{\top}\right]\boldsymbol{L} \\ &\quad where\ A_t = \boldsymbol{p}_t\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{+}\right)^{\top}\end{aligned} \tag{32}$$

Thus, we can train the network using backpropagation or more sophisticated algorithms such as Adams. The steps of Robust-LOSL are summarized in Algorithm3.

Similarly, we can derive the proposed low-rank robust distance learning algorithm named Robust-LODML[13] from the generic optimization problem (26) with the following settings:

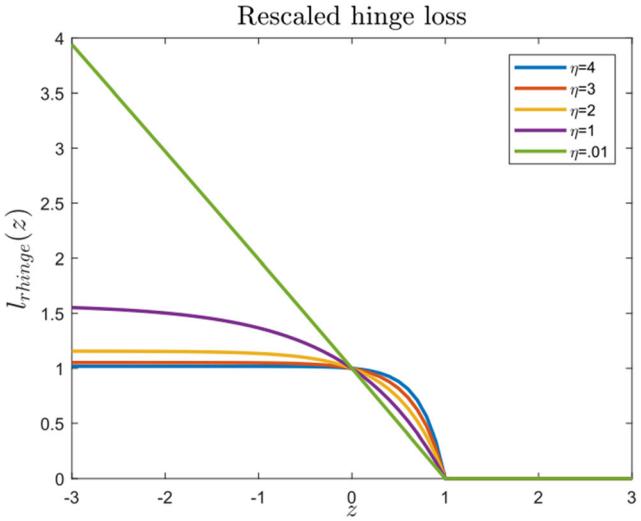

Fig. 3 The Robust Rescaled hinge loss function vs z with different η values

Algorithm3. Robust-LOSL

Inputs: $C, \eta, MaxHQIter$

Output: Similarity transformation matrix $\boldsymbol{L}$

1. Initialize $\boldsymbol{L}$ with the Identity matrix.
2. Set $v = 1$
3. for $t = 1,2,\ldots,T$
   - 3.1. Receive an input triplet $(\boldsymbol{p}_t, \boldsymbol{p}_t^{+}, \boldsymbol{p}_t^{-})$
   - 3.3. for $s = 1,2,\ldots,MaxHQIter$
     - 3.3.1. Update v from (17)
     - 3.3.2 Compute the triplet weight $C_t = C\beta\eta(-v)$.
     - 3.3.3. Optimize the network model parameterized by $\boldsymbol{L}$ using sub-gradient descent algorithm.

   end;

$\mathrm{reg}(\boldsymbol{L}, \boldsymbol{L}_t) = \frac{1}{2}\|\boldsymbol{L}-\boldsymbol{L}_t\|_F^2$, define z_t according to (30).

We solve the obtained problem iteratively by computing v_t from the eq. (17) and then updating $\boldsymbol{L}$ by optimizing the neural network model presented in Fig. 4. The sub-gradient of the loss function with respect to $\boldsymbol{L}$ can be computed from the following equation:

$$\begin{aligned}\frac{\partial l_t}{\partial \boldsymbol{L}}\Big| &= (\boldsymbol{L}-\boldsymbol{L}_t) + 2C_t\left[\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{+}\right)\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{+}\right)^{\top} - \left(\boldsymbol{p}_t-\boldsymbol{p}_t^{-}\right)\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{-}\right)^{\top}\right]\boldsymbol{L} \\ &|= (\boldsymbol{L}-\boldsymbol{L}_t) + 2C_t\boldsymbol{A}_t\boldsymbol{L} \\ &where\ \boldsymbol{A}_t = \left(\boldsymbol{p}_t-\boldsymbol{p}_t^{-}\right)\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{-}\right)^{\top} - \left(\boldsymbol{p}_t-\boldsymbol{p}_t^{+}\right)\left(\boldsymbol{p}_t-\boldsymbol{p}_t^{+}\right)^{\top}\end{aligned} \tag{33}$$

[13] Robust Low-rank Online Distance Metric Learning

Algorithm 4 summarizes the steps of Robust_LODML. We can easily replace the linear module in the proposed low-rank model with a nonlinear deep neural network module. Thus, extending our methods for online deep Distance/Similarity learning is straightforward. Also, the experimental results in the next section confirm that Robust_LODML reduces the computational cost significantly while preserving the predictive performance of the learned metric.

Algorithm4. Robust-LODML

Inputs: $C, \eta, MaxHQIter$

Output: Distance transformation matrix $\boldsymbol{L}$

1. Initialize $\boldsymbol{L}$ with the Identity matrix.
2. Set $v = 1$
3. for $t = 1,2,\ldots,T$
   - 3.1. Receive an input triplet $(\boldsymbol{p}_t, \boldsymbol{p}_t^+, \boldsymbol{p}_t^-)$
   - 3.3. for $s = 1,2,\ldots,MaxHQIter$
     - 3.3.1. Update v from (17)
     - 3.3.2 Compute the triplet weight $C_t = C\beta\eta(-v)$.
     - 3.3.3. Optimize the network model parameterized by L using sub-gradient descent algorithm.

   end;

3.5 Convergence Analysis

This subsection establishes the convergence of our methods with a similar analysis in [9]. According to (14),

$$f(\boldsymbol{M}, \boldsymbol{v}) = -\text{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\left(\eta l_{hinge}(z_t)v_t - g(v_t)\right) = -\text{reg}(\boldsymbol{M}, \boldsymbol{M}_t) + C\beta\exp\left(-\eta l_{hinge}(z_t)\right) \leq C\beta$$

The inequality holds since $\text{reg}(\boldsymbol{M}, \boldsymbol{M}_t) \geq 0$ and $\exp(-\eta l_{hinge}(z_t)) \leq 1$. Thus, our objective function $f(\boldsymbol{M}, \boldsymbol{v})$ is upper bounded. Let $f\left(\boldsymbol{M}_t^{(s)}, \boldsymbol{v}_t^{(s)}\right)$ indicates the objective function in the s-th iteration of the HQ loop. According to (16) and (19), we have

$$f\left(\boldsymbol{M}_t^{(s)}, \boldsymbol{v}_t^{(s)}\right) \leq f\left(\boldsymbol{M}_t^{(s)}, \boldsymbol{v}_t^{(s+1)}\right) \leq f\left(\boldsymbol{M}_t^{(s+1)}, \boldsymbol{v}_t^{(s+1)}\right)$$

It means that the sequence $\left\{f\left(\boldsymbol{M}_t^{(s)}, \boldsymbol{v}_t^{(s)}\right), s = 1, 2, \ldots, MaxHQIter\right\}$ generated by our algorithms is nonincreasing. Consequently, by considering the convergence property of gradient descent methods [21], the convergence of the proposed algorithms are established.

3.6 Run Time Analysis

As seen, the proposed robust online Distance/Similarity learning model is general and can easily be applied to the existing online Distance/Similarity algorithms. Let A be an online Distance/Similarity algorithm with the time complexity T_A. By applying our method to A, besides optimizing the Distance/Similarity measure, we require to compute the weight of the incoming triplet (C_t) using the eq. (17). As seen, C_t requires evaluation of $l_{hinge}(z_t)$ which is also needed for updating the metric. Therefore, it does not imply additional costs, and the overall time complexity of the robust method will be $O(MAXHQIter \times T_A)$. The experimental results confirm that the convergence of the alternating loop is fast, and the best results are obtained by taking $MAXHQIter \leq 3$ in all experiments. Therefore, the obtained robust method has the same time complexity as the corresponding algorithm (A).

3.7 Online Triplet Constructing Algorithm

Generating triplets using available batch algorithms is both time and space consuming. Also, the one-pass triplet constructing strategy adopted in OPML has low performance, especially in noisy environments. To this end, we propose an online triplet constructing algorithm named OCTG[14] which

[14] Online Cluster-based Triplet Generator

is not only very efficient but also effective in comparison with the available batch methods. By utilizing the distribution and clusters of input data, the proposed algorithm can effectively detect outliers and noisy labeled data. Therefore, its performance surpasses existing methods in noisy environments.

Suppose $\{\boldsymbol{V}_i | \ i = 1, 2, \ldots, K\}$ is the set of cluster centers initialized by a sample of data at the beginning of the online algorithm. Here, we use the k-means algorithm to obtain c cluster centers per class in the dataset. OCTG receives incoming data $(\boldsymbol{x}_t, y_t)$ at time step t and finds its closest cluster center $\boldsymbol{V}_t$ with the same class. Then, it considers any cluster center $\boldsymbol{V}_i$ from a different class (i.e., $y_i \neq y_t$) which *violates* the following condition as an *imposter* (see Fig. 5):

$$d(\boldsymbol{x}_t, \boldsymbol{V}_t) + margin < d(\boldsymbol{x}_t, \boldsymbol{V}_i)$$

The triplet set constructed at time step t is formed as:

$$T_t = \{(\boldsymbol{x}_t, \boldsymbol{V}_t, \boldsymbol{V}_i) | \text{ where } \boldsymbol{V}_i \text{ is an imposter } \}$$

As seen, the proposed methods assign a weight denoted by C_t to each incoming triplet. We assign weight w_t to $\boldsymbol{x}_t$ equal to the minimum weights of the generated triplets at time t. The small value for w_t means that $\boldsymbol{x}_t$ is a potential outlier or a noisy labeled instance. The weight and input data are then used to update the cluster centers using any existing online clustering methods.

The obtained weights can be used to enhance the performance of any metric-based algorithms such as kNN or CBIR (Content-Based Information Retrieval) in noisy environments. For example, we use the following version of kNN named Robust-kNN (instead of the standard kNN) to classify the objects in the experiments.

Figure 6 depicts the system flow of the proposed learning/test schemes.

4 Experimental Results

This section deals with the experiments performed to evaluate the performance of proposed methods in noisy environments. First, we study the effect of label noise on the generated triplets and then discuss how these noisy triplets affect the performance of online DML methods. Subsequently, we evaluate the performance of proposed methods on real datasets at different levels of label noise. The results are compared with peer methods.

The Bilinear or dot product similarity learning is equivalent to Mahalanobis distance learning when each instance of the input triplet has a unit norm (i.e., $\|\boldsymbol{p}\|_2 = 1$). Thus, the experiments focus on the Mahalanobis distance learning.

4.1 Effect of Label Noise on the Generated Triplets

As depicted in Fig. 7, we can distinguish between three different types of noisy triplets: *anchor*, *positive*, and *negative noisy triplets*.

To study the effects of different types of noisy triplets, we apply 10% label noise to the Wine dataset. The noisy dataset is visualized using the T-SNE algorithm [22] in Fig. 8.

The statistics of the generated triplets using both batch [17] and OCTG methods are summarized in Table 3.

As the results in Table 3 indicate, by applying only 10% label noise, 68% and 46% of generated triplets by the batch

Algorithm5. Robust-kNN

Inputs:

$\boldsymbol{X}$: training examples

$\boldsymbol{X}_{test}$: test instances

k: k parameter in kNN

$\boldsymbol{w}$: instance weights vector

η: Percentage of noisy labels

1. Sort data according to $\boldsymbol{w}$ values
2. Remove η percent of data in $\mathbf{X}$ with the lowest weights.
3. **for each** instance $\boldsymbol{x}_{test}$ in $\boldsymbol{X}_{test}$
   - 3.1 Compute k nearest neighbors in Set $\boldsymbol{X}$
   - 3.2 Predict label of $\boldsymbol{x}_{test}$ based on the weighted majority voting of its neighbors.

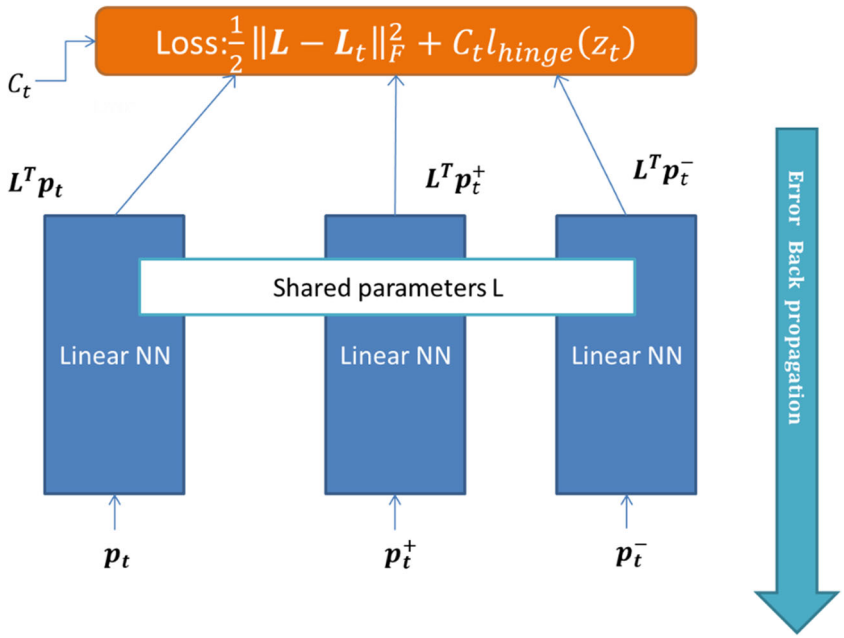

Fig. 4 The proposed neural network model for Low-rank Robust Online Distance/ Similarity learning

and OPML triplet construction methods are contaminated respectively. On the other hand, OCTG only constructs 25% contaminated triplets (just from anchor noisy type). It is due to the fact that OCTG selects positive and anchor points from cluster centers, not data instances that may have been contaminated by label noise. The generated noisy triplets by OCTG have large losses in comparison with that of normal ones (1.67 *vs* 0.39). It can be explained by the fact that a labeled noise example is often far away from its cluster center while it is close to a center from other classes. Hence, the proposed robust methods assign very small weights ($C_t = C\beta\eta \exp(-\eta l_{hinge}(z_t))$) to them in the learning process and so they have a negligible effect on the learned metric.

To analyze the effect of different types of triplet noise in a typical DML task, we run the ODML [4] with the following settings on the generated triplets by the batch method.

ODML: The ODML algorithm.

Ideal ODML: The ideal algorithm which knows the noisy triplets in advance and so ignores them in the training process.

Anchor Ideal ODML: The ideal algorithm that only knows the anchor noisy triplets in advance.

Pos Ideal ODML: The ideal algorithm that only knows the positive noisy triplets in advance.

Neg Ideal ODML: The ideal algorithm that only knows the negative noisy triplets in advance.

In this experiment, we divide the dataset into train/test with a 70/30 ratio and run the above algorithms ten times on the dataset. Figure 9 depicts the mean of obtained results by various algorithms.

For small values of C, the results indicate that the learned metric by ODML has no meaningful difference with that of Euclidean. For large values of C, ODML performs worse than Euclidean, and its accuracy substantially degrades in this noisy environment. Also, among the ideal methods (cannot be implemented in practice), the *Anchor Ideal ODML* has the same performance as Ideal ODML, and others (*Pos Ideal ODML*, *Neg Ideal ODML*)

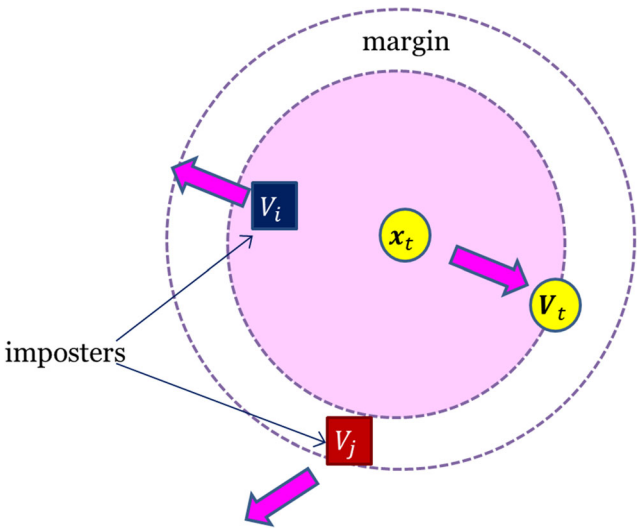

Fig. 5 Illustration of imposters of the data point x_t

are ineffective. Thus, anchor noisy triplets are the main reason for low performance in this experiment.

We repeat the experiment by running Robust-LODML using the triplets generated by our mechanism. The mean-accuracy of kNN-Robust-LODML[15] ($k = 3$, $\eta = 3$) and the weights assigned to instances by Robust-LODML are depicted in Figs. 10 and 11 respectively. As the results show, the proposed method is robust against label noise and its performance surpasses Euclidean metric even for the large values of C. Also, Robust-LODML effectively identified the contaminated instances and considerably reduces their weights (C_t) in the training process.

As shown in Fig. 3, the parameter η controls the robustness of the loss function against outliers and data with noisy labels. To study its effect on the noisy data in a real experiment, we apply 20% label noise on the Wine dataset. Then, we evaluate Robust-LODML in a 5-fold cross-validation setting. Figure 12 depicts the mean accuracy of kNN-Robust-LODML (k = 3) on the dataset. As the result show, the lower η values considerably degrade the performance of Robust-LODML. Also, by properly setting the η value, the performance of our method substantially increases in the noisy environment.

The results are obtained by using only one dataset. In the next subsections, we evaluate the proposed methods on the variety of datasets in different label noise levels. Also, the results are compared with state-of-the-art methods.

4.2 Experimental Setup

Table 4 summarizes the statistics of evaluated datasets in the experiments. Here, all datasets except *Letters* are normalized so that the mean and standard deviation of each attribute becomes 0 and 1, respectively. Also, the dimension of images in *Extended Yale Faces* has been reduced to 100 by applying PCA to alleviate the feature noise effects. The parameter d in Table 4 denotes the input dimension after feature reduction.

In the experiments, triplet side information is generated using OCTG for the proposed methods whereas the one-pass triplet construction [7] is adopted for the other methods.

The results are obtained by k-fold cross validation (k = 5 is set for Letters and Extended Yale Faces and k = 10 for other datasets). The results are compared with peer distance-based methods: ODML [4], LPA-ODML[16] [6], and OPML [7].

The hyperparameters of the competing methods are adjusted by k-fold cross-validation as follows. The parameter C in ODML and the proposed methods are selected from (10^{-6}, 30). The η in the proposed methods is chosen from the range (0.01, 5). Also, λ in OPML is selected from (10^{-6}, 0.05). We evaluate the performance of the learned metrics by the kNN classifier with $k = 3$ in the experiments.

4.3 Results and Analysis

Table 5 presents the classification accuracy of the kNN using the learned metrics of the competing methods. Here, the parameter *nl* shows label noise level (in percent). Figure 7 depicts the mean of 5-fold cross validation accuracy of competing methods versus *nl* (ranging from 0% to 20%). To make the comparison meaningful, the statistical analysis test with $p - value = 5\%$ was performed on the results. In Table 5, we marked our results by * when differences with other methods were statistically significant. Also, boxplots of some statistically different results are depicted in Fig. 14.

As the results in Table 5 and Fig. 13 indicate, the proposed robust methods (i.e., Robust-ODML and Robust-LODML) significantly outperform other DML methods in the presence of label noise. Also, the performance of these methods declines more slowly than other ones with the increase of noise level. That confirms our claim that using the robust loss function and robust sampling preserves the discrimination of the learned metric in a noisy environment.

Besides, the low-rank version of the proposed method (i.e., Robust-LODML) almost has the same accuracy as Robust-ODML. That confirms in real datasets, data lie on a latent subspace with dimensionality $r \ll d$. Thus, learning the projection matrix $L_{d \times r}$ instead of full Mahalanobis matrix $\boldsymbol{M}$ results in the same performance while it is more efficient in terms of time and space requirements.

In the next subsection, we evaluate our proposed methods in a more challenging dataset for identifying COVID-19 patients from Chest-X-ray images.

4.4 Detecting COVID-19 Patients from Chest-X-ray images

4.4.1 Dataset description

The dataset used in our experiments is publicly available in the kaggle repository[17] [25]. Figure 15 depicts some examples from both classes. It contains 219 COVID-19 cases and 1341 normal images. As seen, the dataset is imbalanced and is too small to train a deep CNN model from scratch.

4.4.2 Experimental setup

To extract features from the images, we use the pretrained Resnet18 [26]. This network was trained on the ImageNet

[15] The kNN classifier using the learned metric of Robust-LODML method.
[16] Local Passive/Aggressive Online Distance Metric Learning

[17] https://www.kaggle.com/tawsifurrahman/covid19-radiography-database?select=COVID-19+Radiography+Database

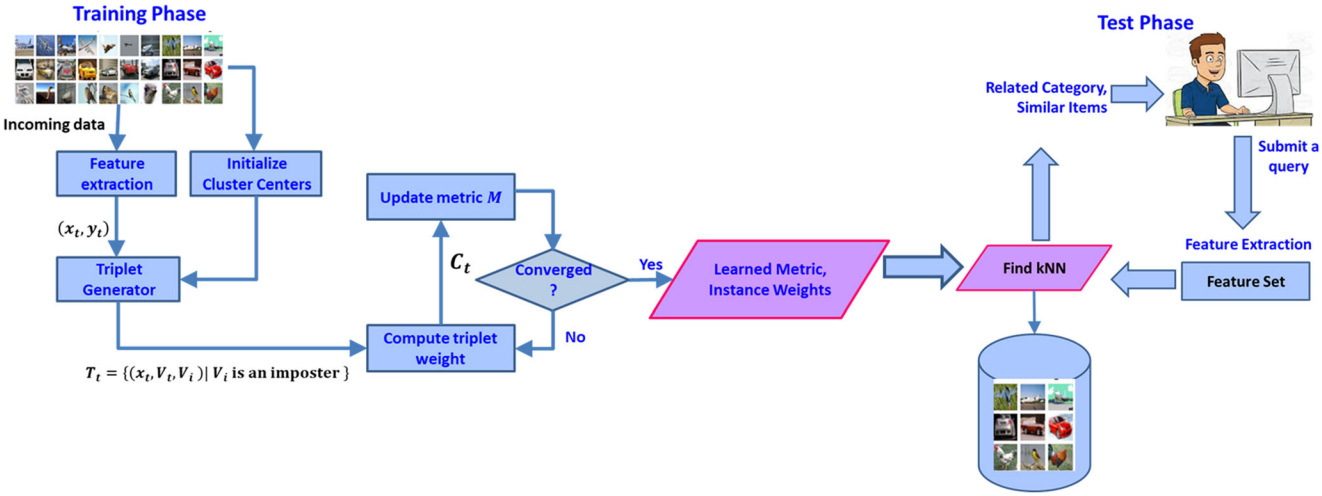

Fig. 6 The system flow of the proposed learning/test schemes

dataset (with 1.4 million labeled images and 1000 different classes). It has 71 layers, and the input layer requires images of size 224-by-224-by-3. We resize the images to the specified size and obtain 512 features from the global pooling layer, *'pool5'*, at the end of the model.

In addition of online methods, we also compared the proposed methods with the BLMNN [18] batch method. The λ and *maxIter* hyperparameters of BLMNN are selected from the ranges {1, 3, 5, 10, 20} and {1, 3, 5} respectively using 5-fold cross-validation.

We use 5-fold cross-validation to obtain the results in the experiments. The main concern in this task is to limit the number of missed COVID-19 cases. Hence, in addition to accuracy, we utilize a variety of metrics to evaluate our work. These metrics are Sensitivity (Recall), Precision, F1 Score, and G-mean (Geometric-mean). Here, COVID-19 and Normal are considered as positive and negative classes, respectively. The metrics are defined as follows:

$$Accuracy = (TP + TN)/All\ Predictions \tag{34}$$

$$Sensitivity\ (Recall) = TP/(FN + TP) \tag{35}$$

$$Precision = TP/(TP + FP) \tag{36}$$

$$F1{-}Score = 2\ (Precision \times Sensitivity)/(Precision + Sensitivity) \tag{37}$$

$$Specificity = TN/(TN + FP) \tag{38}$$

$$G{-}mean = \sqrt{Sensitivity \times Specificity} \tag{39}$$

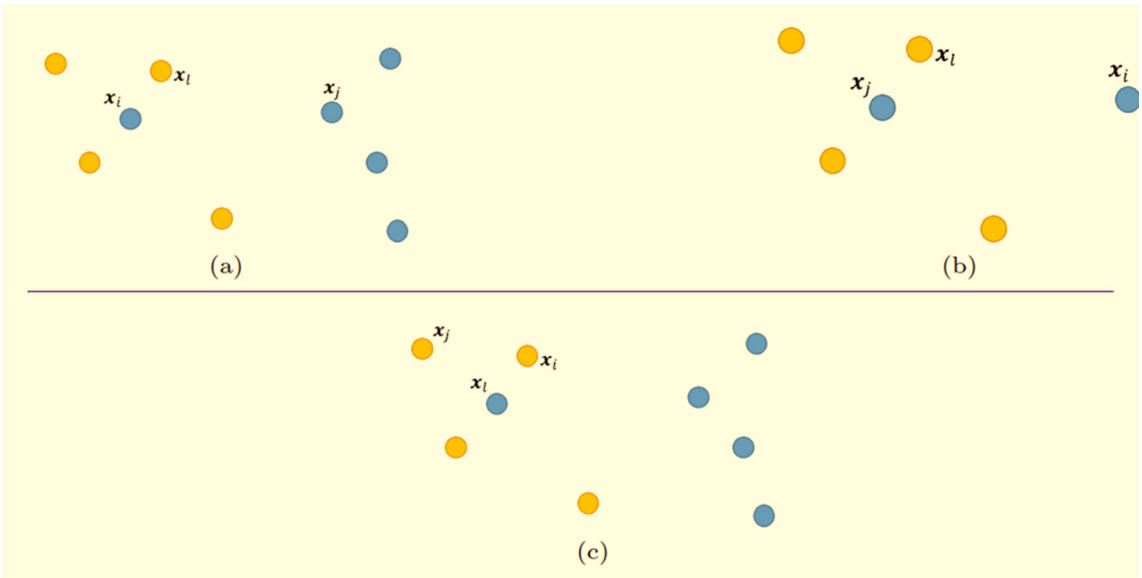

Fig. 7 Three different types of noisy triplets in the form (x_i, x_j, x_l): (**a**) Anchor noisy triplet where x_i is contaminated with label noise, (**b**) Positive noisy triplet where x_j has label noise, and (**c**) Negative noisy triplet where x_l has a wrong label

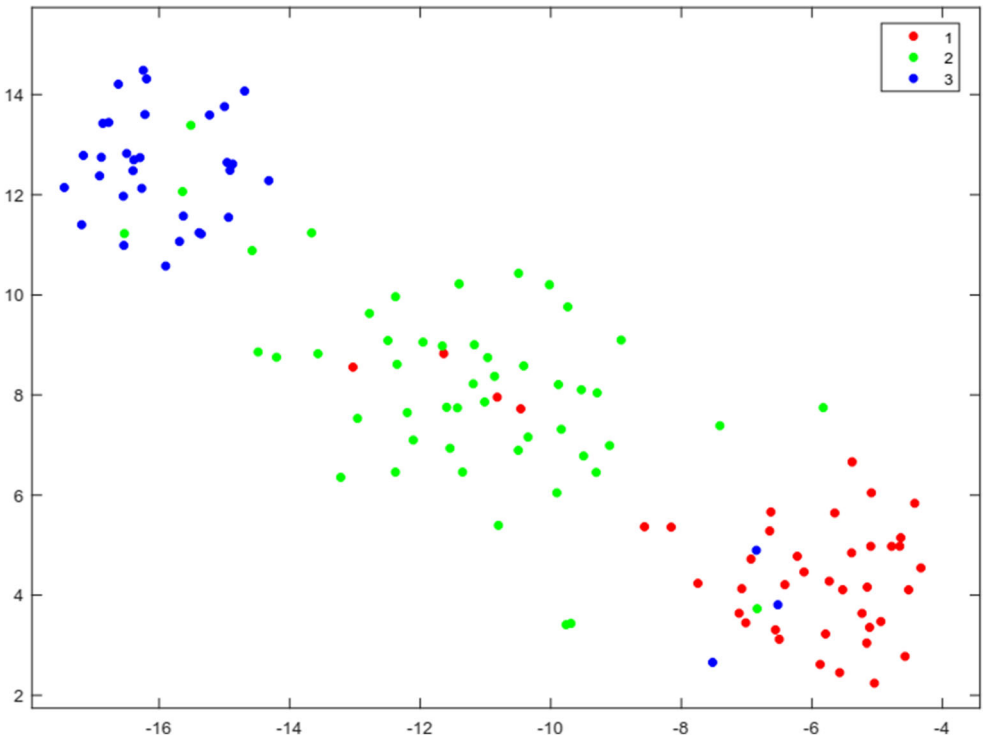

Fig. 8 T-SNE Visualization of the Wine dataset after applying 10% label noise

4.4.3 Results and analysis

Table 6 presents the classification results of the kNN using the learned metrics in the different levels of label noise. The results of both *sensitivity* and *precision* of the competing methods versus noise level are shown in Fig. 16(a). Since sensitivity is more important in this task, we multiply it by 2. Also, Fig. 16(b) presents the G-mean results versus noise level. The high value of G-mean indicates that accuracy in both classes is high and balanced.

As the results indicate, all methods obtain a high performance in a low-level label noise setting. However, with the increase of noise level, the performance of the competing methods declines sharper than our proposed methods. Especially, while the BLMNN (batch method) has the advantage of processing each data multiple times, it does not perform well in high-level noise settings. It can be explained as follows: 1) the batch triplet sampling utilized in BLMNN is vulnerable to label noise as discussed in subsection 4.1, 2) while Bayesian learning is effective to deal with feature noise, it is less helpful to deal with the more complicated problem (i.e., label noise).

The proposed methods achieve high sensitivity for COVID-19 patients in noisy environments. It is very important since the primary goal of this task is to limit the

Table 3 Statistics of generated triplets in the Wine dataset contaminated with 10% label noise

Method	Batch		OPML		OCTG (Ours)	
Feature	#	Mean Hinge loss	#	Mean Hinge loss	#	Mean Hinge loss
Instances	178	–	178	–	178	–
Classes	3	–	3	–	3	–
Triplets	413	0.92	85		140	0.71
Normal triplets	131	0.85	46	0.45	105	0.39
Noisy triplets	282	0.96	39	0.51	35	1.67
Anchor noisy triplets	38	1.01	17	0.57	35	1.67
Positive noisy triplets	23	1.02	17	0.51	0	–
Negative noisy triplets	249	0.95	14	0.42	0	–

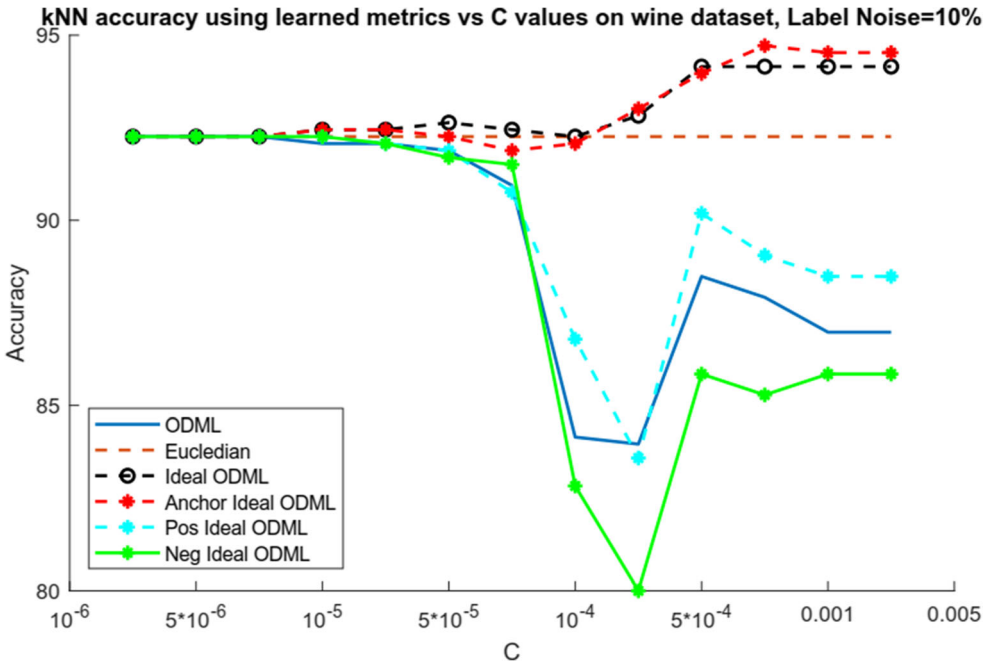

Fig. 9 The kNN (k = 3) accuracy of the learned metric of various algorithms in the Wine dataset with 10% label noise

number of misclassified COVID-19 cases as much as possible. For example, the Confusion matrices of the proposed methods at *Noise Level* = 20% are shown in Table 7. As seen, only 1.8 and 1 (as the average of 5-fold cross validation) COVID-19 patients are misclassified as Normal by the proposed methods. Also, our methods obtain good precision (or predictive positive value). High precision is crucial since high FP (False Positive) increases the burden of the healthcare system for additional care and tests such as PCR (Polymerase Chain Reaction). Therefore, based on the results, we can conclude the proposed methods perform well in detecting COVID-19 cases in the presence of label noise. However, the difference between *sensitivity* and *specificity* values indicates further improvements are possible by adopting *balancing techniques* in this imbalanced dataset.

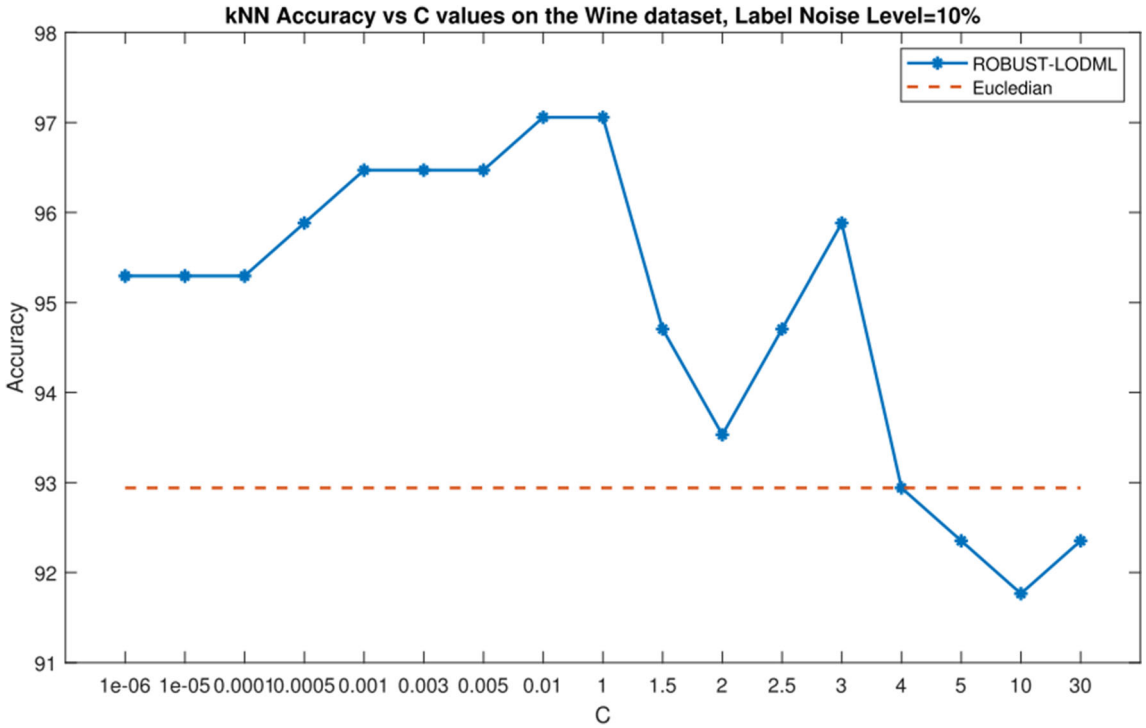

Fig. 10 The kNN accuracy of the learned metric by Robust-LODML algorithm ($\eta = 3$) in the Wine dataset with 10% label noise

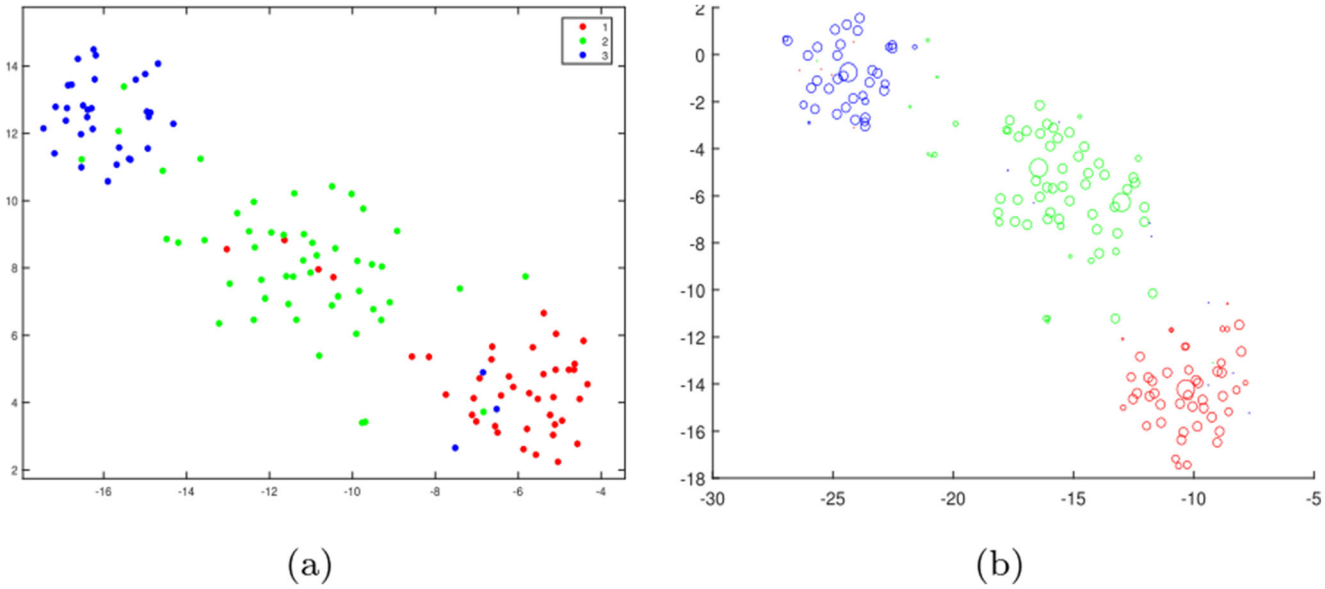

Fig. 11 The tSNE visualization of the Wine dataset with 10% label noise where data points are displayed (**a**) with equal sizes (**b**) with the sizes proportional to their weights

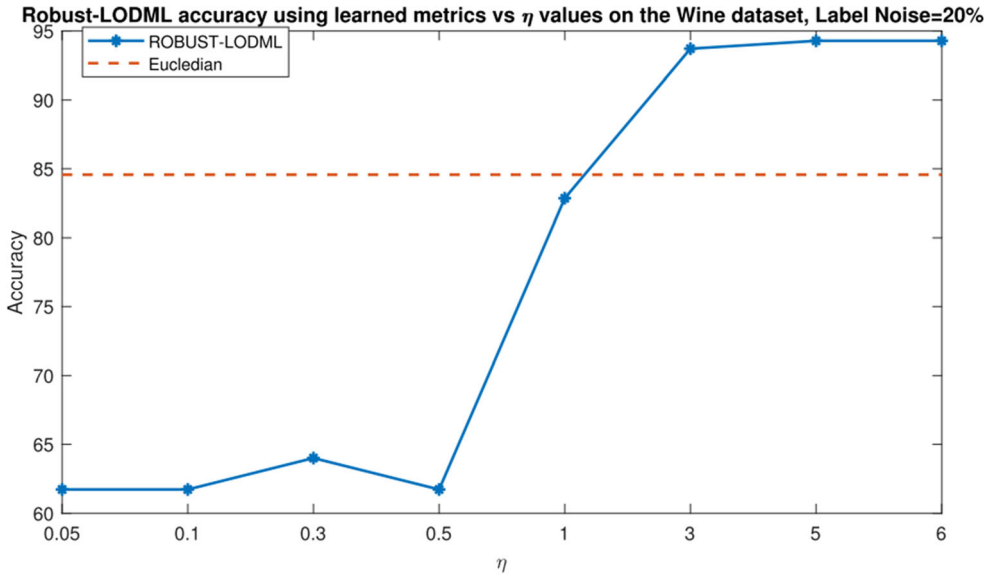

Fig. 12 Mean accuracy of kNN-Robust-LODML (k = 3) vs. η values on the Wine dataset with 20% label noise

Also, we studied the mean run time of the competing methods in a 5-fold cross-validation setting. The results are depicted in Fig. 17. Also, Table 8 shows the summary of statistics in the experiment. Here, in the "hyper-parameters" column, we only report the value of time-related hyper-parameters. The parameter r indicates the number of columns in the projection matrix $\boldsymbol{L} \in \mathbb{R}^{d \times r}$. Note that, OPML only can learn a square projection matrix (r = d = 512 in these experiments), while Robust-LODML can learn a rectangular low-rank matrix. For Robust-LODML, we adjust the value of r from {128, 256, 512}. The #active column shows the mean number of active triplets.[18] It also indicates the number of times that the algorithm should update the metric.

The overall execution time of a DML method depends on the efficiency of the triplet sampling mechanism, the required time to update the metric, and its convergence rate. In the noise-free experiment, the average number of generated triplets by the one-pass triplet construction algorithm is 1231 (refer to Table 8). However, only a few of them violate the margin constraint. The mean number of active triplets for LPA-ODML, ODML, and OPML are 65.00, 100.60, and 33.20, respectively. Thus, OPML achieves a low runtime in this experiment. On the other hand, the OCTG utilized in our methods only generates an average of 43.40 triplets. The

[18] Triplets that violate the margin and so have none zero loss.

Table 4 Statistics and explanations of evaluated datasets

Data Set	#classes	n	#dim	d	Description
Wine [23]	3	178	13	13	Standard UCI classification dataset. https://archive.ics.uci.edu/ml/datasets/wine
Letters [23]	26	20,000	16	16	includes 20,000 examples of 26 English capital letters. Images of letters are generated from 20 different fonts and then 16 numerical attributes are extracted from these images. https://archive.ics.uci.edu/ml/datasets/letter+recognition
Extended Yale Faces [24]	38	2414	1024	200	is a standard face recognition dataset that contains 2414 face images of 38 classes. For each person, at most 64 images are taken under extreme illumination conditions. http://vision.ucsd.edu/~iskwak/ExtYaleDatabase/ExtYaleB.html
Ionosphere [23]	10	351	34	33	Standard UCI classification dataset. https://archive.ics.uci.edu/ml/datasets/Ionosphere
WDBC [23]	2	569	32	30	Breast Cancer Wisconsin (Diagnostic) Data Set https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
Australian	2	690	14	14	was used in a competition on click-through rate prediction jointly hosted by Avazu and Kaggle in 2014. The participants were asked to from the first 10 days of advertising log, estimate the click probability for the impressions on the 11th day. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html
German Credit	2	1000	24	24	Each instance represents a person who takes a credit from a bank and is classified as good or bad credit risks according to the set of attributes. https://www.kaggle.com/uciml/german-credit

Table 5 The classification accuracy of the kNN using the learned metric of the competing methods

Data Set	*nl* %	Robust-LODML	Robust-ODML	LPA-ODML	ODML	OPML	Euclidean
Wine	0	97.65±4.11	96.47±6.32	97.06±4.16	**98.29±1.56**	97.06±4.16	95.29±5.41
	5	**97.65±3.04**	97.06±3.10	97.06±5.00	96.00±4.78	95.88±4.84	93.53±5.154
	10	96.47±4.96	**97.06±5.00**	96.47±4.11	93.14±3.26	96.47±4.11	94.12±4.80
	15	**97.65±4.11**	95.88±5.58	94.71±6.47	90.86±6.52	94.12±5.55	92.35±6.82
	20	**95.29±6.68**	**95.29±5.41**	90.00±6.82	89.14±3.73	91.18±8.43	85.88±8.41
Letters	0	96.80±0.25	96.68±0.37	**96.88±0.25**	96.76±0.29	96.78±0.34	95.39±0.36
	5	**96.41±0.35**	95.85±0.61	96.08±0.29	95.98±0.28	96.02±0.37	94.53±.50
	10	**95.39±0.35***	94.36±0.44	93.57±0.34	94.08±0.32	94.29±0.31	92.64±0.51
	15	**94.19±0.40***	93.27±0.63	91.78±0.28	91.53±0.71	91.57±0.40	90.03±0.55
	20	**93.18±0.48***	92.20±1.04	88.69±0.30	88.46±0.40	88.32±0.32	86.67±0.81
Extended Yale Faces	0	**96.02±0.31***	95.52±1.12	93.94±0.90	93.82±.82	93.57±0.88	93.36±0.89
	5	**95.56±0.45***	94.27±1.12	92.86±0.84	92.82±1.05	92.53±0.62	92.57±0.27
	10	**94.94±1.01***	93.69±1.02	92.78±1.24	91.70±1.41	90.95±1.23	91.54±0.86
	15	**94.90±1.20***	92.70±1.78	91.33±0.90	88.51±1.19	88.71±1.32	88.63±1.03
	20	**93.11±1.40***	92.37±0.68	88.88±1.26	85.56±0.91	85.23±1.28	85.56±0.91
Ionosphere	0	93.14±3.35	92.00±3.24	**94.00±4.94**	90.29±4.09	86.57±5.05	84.86±3.29
	5	**93.14±4.09**	91.43±5.39	91.71±4.75	89.71±3.61	87.43±4.30	86.00±3.41
	10	90.86±4.82	**91.43±4.47**	88.86±4.56	87.71±3.82	86.57±3.31	84.57±4.89
	15	**90.00±4.90**	89.14±9.31	87.14±6.06	87.71±5.23	84.35±7.31	81.43±2.26
	20	**88.86±4.75**	88.00±5.35	83.71±4.68	84.29±6.35	82.32±7.25	79.43±6.29
WDBC	0	**95.71±2.41**	95.36±2.55	95.18±3.04	95.00±2.64	95.00±3.84	92.86±3.15
	5	94.29±3.01	**95.54±2.70**	94.82±2.59	93.93±3.17	93.75±3.88	92.32±3.37
	10	**94.46±3.31**	93.57±2.41	93.04±2.59	92.68±2.97	93.57±2.94	89.11±3.81
	15	**94.11±3.04**	92.32±3.67	90.18±3.69	88.39±6.08	88.93±2.35	85.71±5.26
	20	91.07±2.92	**92.50±4.67***	85.89±3.62	85.00±5.60	85.89±3.81	83.93±6.63
Australian	0	85.51±5.11	86.23±3.82	**86.67±4.72**	85.51±3.98	83.62±6.11	82.03±5.43
	5	86.09±5.17	**87.25±5.24**	85.94±5.68	84.93±4.44	83.33±5.08	81.30±5.31
	10	84.78±2.49	**86.67±4.72**	83.77±5.01	81.45±4.62	81.45±4.92	78.99±7.43
	15	**85.07±4.88**	**85.07±4.32**	82.32±4.92	78.26±5.68	81.45±3.54	76.81±5.76
	20	85.51±3.92	**85.80±3.19***	79.42±5.10	74.64±6.84	78.84±4.94	73.19±5.12
German Credit	0	74.60±2.07	74.70±2.21	**76.10±3.60**	73.90±2.38	72.20±3.79	69.40±4.14
	5	**74.60±2.55**	73.40±5.42	73.50±4.40	72.40±4.20	70.30±4.50	67.90±5.26
	10	73.20±4.44	**74.00±3.40**	72.70±3.13	71.10±5.22	69.40±5.21	66.50±5.99
	15	73.20±3.58	**73.30±4.90**	71.20±5.09	68.60±2.95	65.20±3.88	64.30±4.79
	20	**72.60±3.50**	71.50±4.14	70.30±5.54	65.70±3.68	63.30±5.14	62.30±6.20

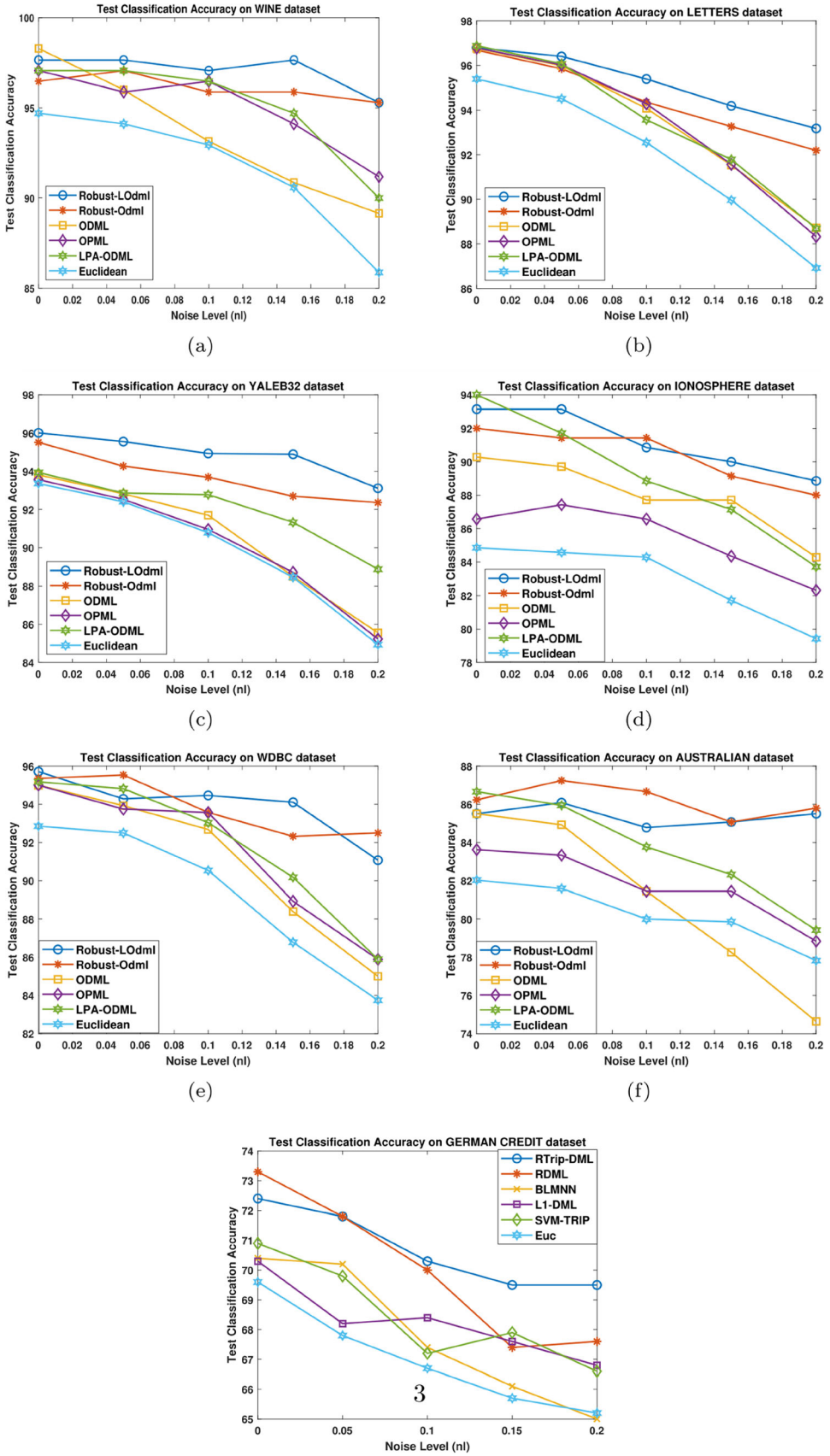

Fig. 13 Comparison of the classification accuracy of RDML with other DML methods versus label noise

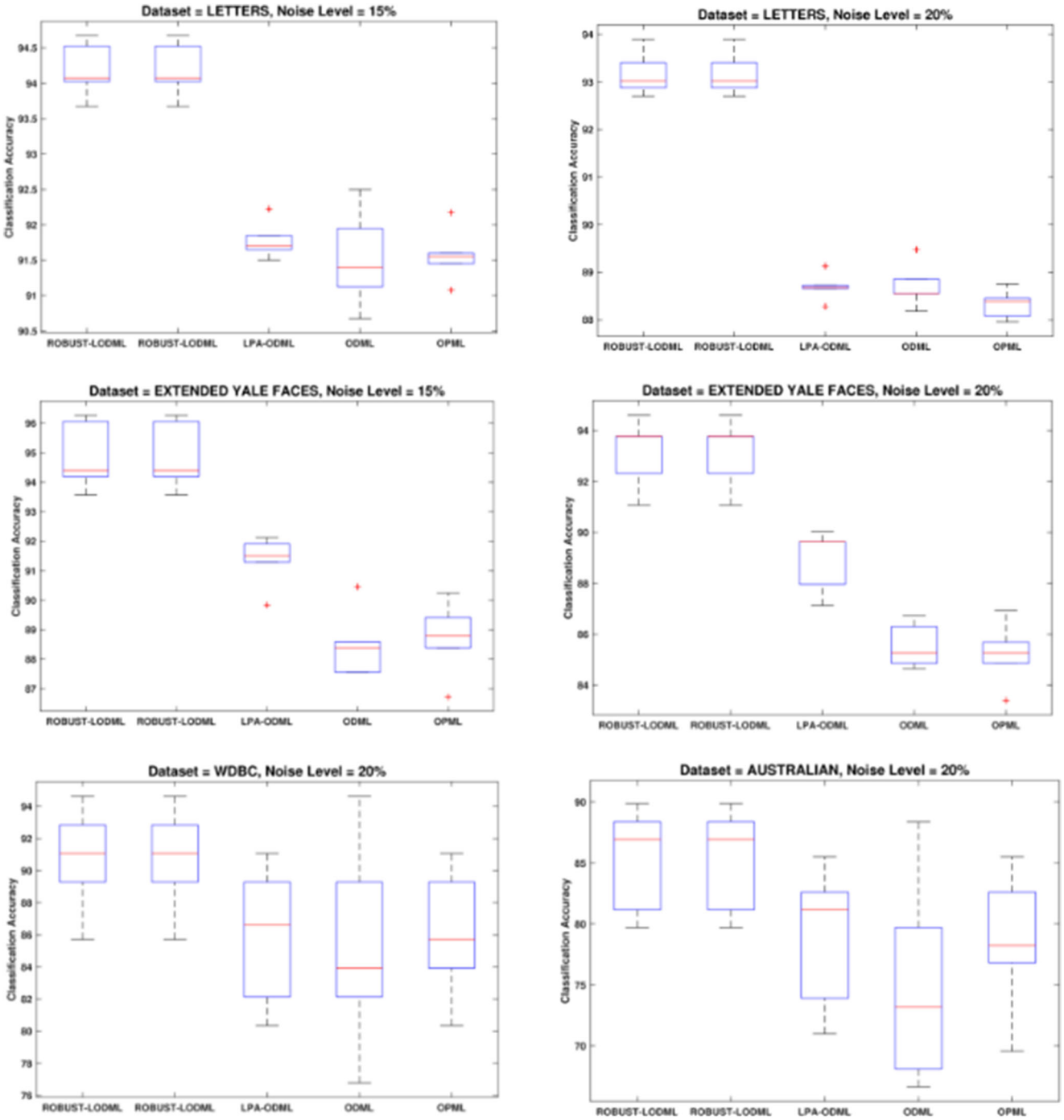

Fig. 14 Boxplots of some statistically different results with p-value = 5%

average number of active triplets for Robust ODML and Robust LODML are 21.00 and 26.80, respectively. As seen, the execution time of both Robust-ODML and Robust-LODML are acceptable in this experiment.

In the high-level noisy environment (nl = 20%), the convergence rate of non-robust methods (i.e., LPA-ODML, ODML, and OPML) is low. Therefore, the number of active constraints is high, and their execution times exceed the robust algorithms. Here, we found that the best hyper-parameters setting for Robust-LODML is r = 128, $MaxHQIter$ = 1. Hence, the number of its parameters is a quarter of other methods. Also, it only has an average of 292.60 active constraints. Thus, its run time is considerably smaller than other competing methods.

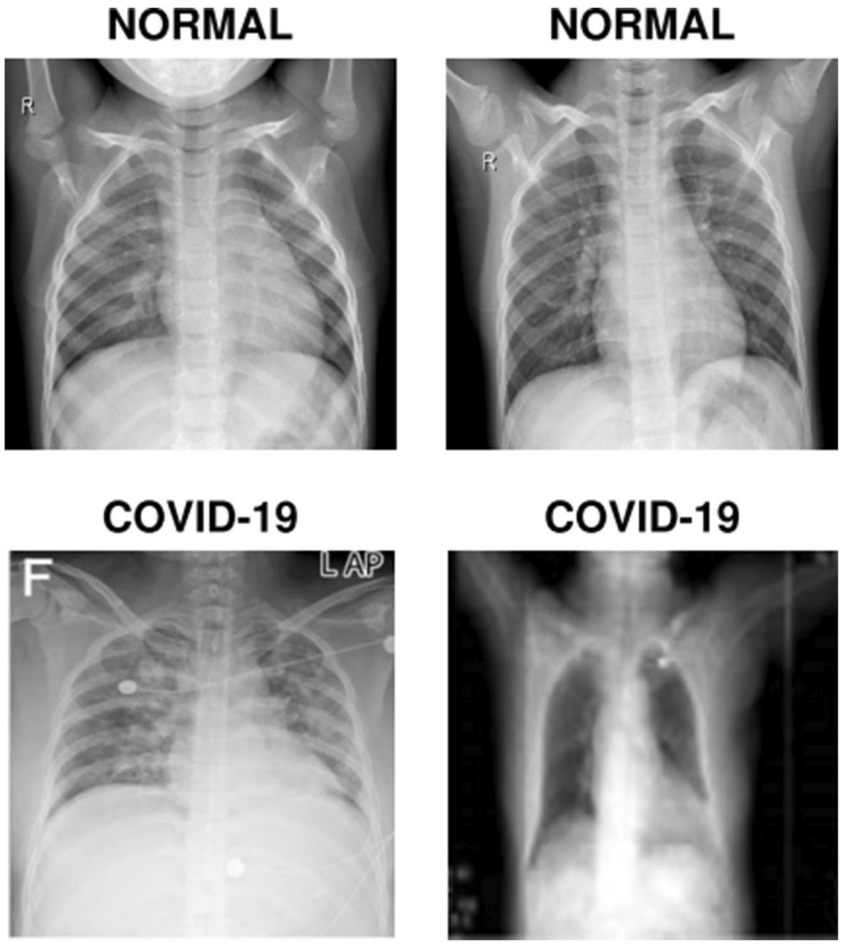

Fig. 15 Four images from the COVID-19 dataset. First row: Normal cases, Second row: COVID-19 patients

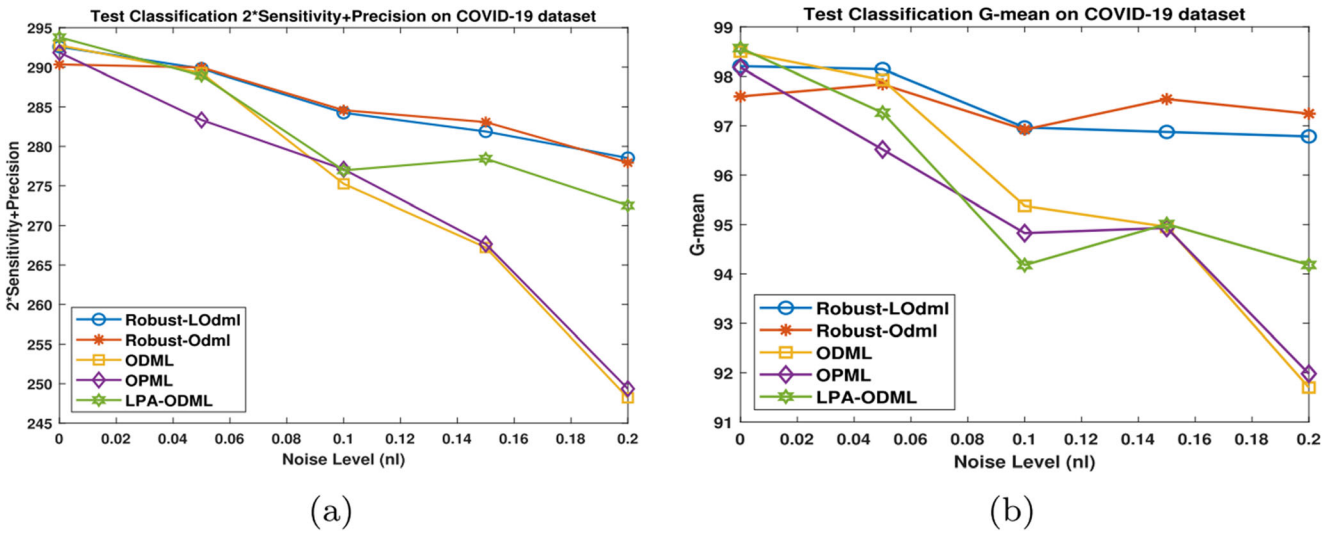

Fig. 16 2×Sensitivity + Precision and G-means of the competing methods on the COVID-19 dataset

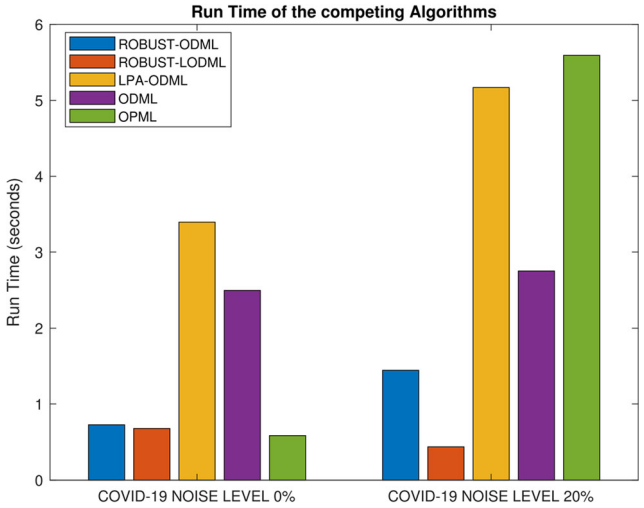

Fig. 17 Mean run time of evaluated methods in 5-fold validation setting in the COVID-19 dataset

5 Conclusion and Future work

Existing online Distance/Similarity learning methods are usually formulated by the Hinge loss and so are not robust against outliers and label noise data. Also, they often have the wrong assumption that training triplets or pairwise constraints exist in advance. Generating triplets using available batch algorithms is both time and space consuming. To address these challenges, we formulate the online Distance/Similarity learning problem using the robust Rescaled Hinge loss [9]. Also, we develop an efficient robust one-pass triplet sampling algorithm that takes data distribution and its clusters into account.

We further extend our work by providing the low-rank variants of proposed methods that learn a rectangular projection matrix instead of a full Mahalanobis matrix.

We studied the effects of label noise in a DML task and conducted several experiments to measure the performance of the proposed methods at different noise levels. Extensive experimental results show that the proposed methods can effectively detect wrong label data and reduce their influences in DML tasks. Thus, they consistently outperform other related online Distance/Similarity learning algorithms in noisy environments.

We intend to extend the work for online deep distance/similarity learning. Some other directions for future work are:

Table 6 Classification metrics of kNN using the learned metrics of competing methods on the COVID-19 dataset

Method	*nl*%	Accuracy	Sensitivity	Precision	Specificity	G-mean	F1-Score
Robust-ODML	0	99.23±0.66	95.35±3.52	**99.65±0.78**	99.92±0.18	97.59±1.83	97.43±1.99
Robust-LODML		99.42±0.57	96.54±3.80	99.50±1.12	**99.93±0.17**	98.20±1.96	97.97±2.16
LPA-ODML		**99.49±0.37**	**97.33±3.06**	99.13±1.25	99.85±0.21	**98.57±1.50**	**98.19±1.32**
ODML		99.36±0.23	**97.33±3.06**	98.10±2.18	99.70±0.31	98.50±1.42	97.66±0.88
OPML		99.29±0.42	96.62±2.61	98.60±2.29	99.78±0.33	98.18±1.25	97.56±1.10
BLMNN		99.23±0.70	96.29±3.77	98.57±0.97	99.93±0.17	97.62±2.10	97.40±1.96
Robust-ODML	5	99.10±0.48	96.11±2.61	97.75±1.30	99.62±0.28	97.84±1.30	96.90±1.13
Robust-LODML		98.97±0.35	**96.99±3.01**	95.83±1.25	99.33±0.17	**98.14±1.47**	96.37±1.08
LPA-ODML		**99.17±0.62**	94.71±4.12	**99.52±1.06**	**99.93±0.17**	97.26±2.13	**97.02±2.32**
ODML		98.97±0.42	96.50±3.37	96.29±2.26	99.40±0.33	97.93±1.64	96.34±1.39
OPML		98.53±0.49	93.81±3.21	95.72±5.45	99.34±0.79	96.52±1.41	94.62±1.97
BLMNN		98.59±0.87	92.64±3.06	97.23±3.73	99.55±0.62	96.02±1.78	94.86±3.02
Robust-ODML	10	**98.46±0.80**	94.90±2.03	94.78±3.34	99.00±1.01	96.92±0.87	**94.79±1.32**
Robust-LODML		**98.46±0.83**	**94.97±2.98**	94.31±3.53	99.02±0.64	**96.96±1.66**	94.62±2.78
LPA-ODML		98.21±1.23	88.95±7.19	**99.07±1.30**	**99.85±0.21**	94.18±3.73	93.59±3.62
ODML		97.50±0.89	92.53±4.39	90.20±7.81	98.37±1.22	95.38±2.00	91.09±3.33
OPML		98.08±0.60	90.53±5.19	96.04±4.68	99.41±0.66	94.83±2.44	93.00±1.19
BLMNN		97.88±1.12	89.90±7.86	94.76±2.82	99.18±0.49	94.35±4.15	92.10±4.48
Robust-ODML	15	97.95±0.66	**96.96±2.46**	89.16±4.22	98.14±0.62	**97.54±1.32**	92.84±2.59
Robust-LODML		97.76±1.96	95.68±3.36	90.52±7.72	98.09±1.80	96.87±2.46	92.97±5.58
LPA-ODML		**98.21±0.98**	90.78±3.68	**96.87±2.13**	**99.47±0.45**	95.01±1.99	**93.70±2.35**
ODML		95.90±0.83	93.69±4.61	79.84±5.70	96.29±0.75	94.96±2.35	86.09±3.79
OPML		96.03±0.18	93.41±2.71	80.84±4.63	96.50±0.48	94.93±1.16	86.56±1.86
BLMNN		96.28±2.03	86.19±10.62	87.38±7.07	97.91±1.21	91.72±5.87	86.57±7.55
Robust-ODML	20	96.73±0.69	**97.94±2.03**	82.06±5.47	96.57±0.93	**97.24±0.82**	89.17±2.67
Robust-LODML		97.31±1.03	96.08±4.67	86.35±5.73	97.54±0.97	96.78±2.38	90.82±3.51
LPA-ODML		**97.50±1.66**	89.82±4.86	**92.88±7.40**	**98.80±1.30**	94.18±2.90	**91.23±5.36**
ODML		92.63±0.93	90.45±3.36	67.32±5.37	92.99±0.71	91.70±1.91	77.09±3.93
OPML		92.44±1.41	91.40±5.26	66.55±7.70	92.64±1.62	91.98±2.59	76.75±5.41
BLMNN		93.91±1.55	81.31±7.12	76.70±4.70	95.97±0.81	88.27±4.19	78.88±5.55

Table 7 Mean of confusion matrices of proposed methods obtained by 5-fold cross validation on the COVID-19 dataset (label noise = 20%)

		Predicted Positive (COVID-19)	Predicted Negative (Normal)
Robust-LODML	Actual Positive (COVID-19)	42.00	1.8
	Actual Negative (Normal)	6.60	261.60
Robust-ODML	Actual Positive (COVID-19)	42.8	1.00
	Actual Negative (Normal)	9.2	259.00

Table 8 Summary of statistics and run-time of the competing methods in a noise free ($nl = 0\%$) and high-level noisy ($nl = 20\%$) settings

Method	*nl%*	#triplets	#active	hyper-parameters	run-time (sec)
Robust-ODML	0	43.40	21.00	*MaxHQIter*=3	0.7297
Robust-LODML		43.40	26.80	*r*=256, *MaxHQIter*=3	0.6810
LPA-ODML		1231.00	65.00	-	3.3945
ODML		1231.00	100.60	-	2.4982
OPML		1231.00	33.20	*r*=512	**0.5871**
Robust-ODML	20	323.00	325.80	*MaxHQIter*=3	1.4442
Robust-LODML		323.00	292.60	*r*=128, *MaxHQIter*=1	**0.4386**
LPA-ODML		1245.00	508.80	-	5.1661
ODML		1245.00	572.40	-	2.7546
OPML		1245.00	515.00	*r*=512	5.5909

I. Examining the performance of the proposed methods in other applications like *CBIR*.
II. Extension of the proposed methods in *imbalanced* environments.
III. Enhance the performance of the proposed online triplet sampling algorithm.

Acknowledgments We would like to acknowledge the Machine Learning Lab in the Engineering Faculty of FUM for their kind and technical support.

References

1. Bellet A, Habrard A, Sebban M (2013) A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv: 1306.6709
2. Chechik G, Sharma V, Shalit U, Bengio S (2010) Large scale online learning of image similarity through ranking. J Mach Learn Res 11: 1109–1135
3. Xia H, Hoi SC, Jin R, Zhao P (2014) Online multiple kernel similarity learning for visual search. IEEE Trans Pattern Anal Mach Intell 36(3):536–549
4. Wu P, Hoi SC, Zhao P, Miao C, Liu Z-Y (2016) Online multi-modal distance metric learning with application to image retrieval. IEEE Trans Knowl Data Eng 28(2):454–467
5. Zhong G, Zheng Y, Li S, Fu Y (2017) SLMOML: online metric learning with global convergence. IEEE Trans Circuits Syst Video Technol 28(10):2460–2472
6. Hamdan B, Zabihzadeh D (2021) Large-Scale Local Online Similarity/Distance Learning Framework Based on Passive/ Aggressive. *Int J Pattern Recognit Artif Intell* 35:2151017
7. Li W, Gao Y, Wang L, Zhou L, Huo J, Shi Y (2018) OPML: a one-pass closed-form solution for online metric learning. Pattern Recogn 75:302–314
8. Rasheed AS, Zabihzadeh D, Al-Obaidi SAR (2020) Large-Scale Multi-modal Distance Metric Learning with Application to Content-Based Information Retrieval and Image Classification. *Int J Pattern Recognit Artif Intell* 34(13):2050034
9. Xu G, Cao Z, Hu B-G, Principe JC (2017) Robust support vector machines based on the rescaled hinge loss function. Pattern Recogn 63:139–148
10. Kaya M, Bilge HŞ (2019) Deep metric learning: A survey. *Symmetry* 11(9):1066
11. Zabihzadeh D, Monsefi R, Yazdi HS (2018) Sparse Bayesian similarity learning based on posterior distribution of data. Eng Appl Artif Intell 67:173–186
12. Qian Q (2015) Large-scale high dimensional distance metric learning and its application to computer vision. Michigan State University. Computer Science-Doctor of Philosophy
13. Davis JV, Kulis B, Jain P, Sra S, Dhillon IS (2007) "Information-theoretic metric learning," presented at the proceedings of the 24th international conference on machine learning, Corvalis, Oregon, USA
14. Gao Y, Li Y-F, Chandra S, Khan L, Thuraisingham B (2019) Towards self-adaptive metric learning on the fly. in *The World Wide Web Conference*, pp. 503–513
15. Yang T, Jin R, Jain AK (2010) Learning from Noisy Side Information by Generalized Maximum Entropy Model. in *ICML*, pp. 1199–1206
16. Zabihzadeh D, Monsefi R, Yazdi HS (2019) Sparse Bayesian approach for metric learning in latent space. Knowl-Based Syst 178: 11–24

17. Weinberger KQ, Saul LK (2009) Distance metric learning for large margin nearest neighbor classification. *J Mach Learn Res* 10, no. Feb:207–244
18. Wang D, Tan X (2018) Robust distance metric learning via Bayesian inference. IEEE Trans Image Process 27(3):1542–1553
19. Boyd S, Boyd SP, Vandenberghe L (2004) Convex optimization. Cambridge University press
20. Xue X, Zhang X, Feng X, Sun H, Chen W, Liu Z (2020) Robust subspace clustering based on non-convex low-rank approximation and adaptive kernel. Inf Sci 513:190–205
21. Shapiro A, Wardi Y (1996) Convergence analysis of gradient descent stochastic algorithms. J Optim Theory Appl 91(2):439–454
22. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. Journal of Machine Learning Research 9(11):2579−2605
23. Dua D, Graff C (2019) UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science
24. Lee K-C, Ho J, Kriegman DJ (2005) Acquiring linear subspaces for face recognition under variable lighting. IEEE Trans Pattern Anal Mach Intell 27(5):684–698
25. Chowdhury ME et al (2020) Can AI help in screening viral and COVID-19 pneumonia? IEEE Access 8:132665–132676
26. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778

Davood Zabihzadeh currently is an assistant professor in computerengineering department of Hakim Sabzevari University (HSU). He receives hisPh.D. in AI from Ferdowsi University of Mashhad. His current research interestsinclude deep metric learning, zero and few shot learning, machine vision, anddeep neural networks.

Amar Tauma Hamdan received the M.S. degree from FerdowsiUniversity of Mashhad in 2019. Currently, she is a Ph.D. candidate in Isfahanuniversity. She is current research interests include deep metriclearning, and deep neural networks.

Ali Karami-Mollaee currently is an associate professor in Electricaland Computer Engineering Faculty of Hakim Sabzevari University (HSU). Hereceives his Ph.D. in Control engineering from Ferdowsi University of Mashhad.His current research interests include nonlinear control and fuzzy logic.

Seyed Jalaleddin Mousavirad gained his PhD in Computer Engineering, Artificial Intelligence from the University of Kashan, Iran. He worked at theUniversity of Tehran (2018-2019), in the School of Computer Engineering, AzadUniversity (2019-2020), in the School of Engineering, before joining theComputer Engineering Department at Hakim Sabzevari University. He has anoutstanding research record and significant capabilities in the area of patternrecognition and machine learning, image processing, and evolutionarycomputation.