Responses to Editors' and Reviewers' Comments

We have completed the major revision for "Large Margin Weighted k-Nearest Neighbors Label Distribution Learning for Classification" (Manuscript ID: TNNLS-2022-P-24063). We sincerely thank the editors and the reviewers for their valuable comments, which are indeed helpful to improve our manuscript.

First, our responses to the comments of the editors are as follows.

Senior Editor 1:

Comments: The paper has received concordant review highlighting a potentially interesting work for the community. However, there are major issues well summarized in the AE's and reviewers' reports that need to be carefully taken into consideration before this work can be accepted for publication.

Response: Thanks for the valuable feedback. We have made significant revision according to the AE's and reviewers' comments.

Associate Editor 2:

C1: In view of referee comments, I am unable to recommend acceptance of this paper in its present form, however, authors may wish to resubmit a new version, considering referee comments and respond to all their queries.

R1: We have made significant improvements according to the reviewers' comments. The major changes mainly include:

1) We have rewritten the sixth paragraph of Section I and the second paragraph of Section II-C to better highlight our novelties in this study.
2) We have added more comparative experiments, a new figure (Fig. 5), and two new paragraphs in Section VI-B.
3) We have rewritten Section VI-C2, redrawn Figs. 7-9, and added a new table (Table VII) to better justify the advantages of large margin.
4) We have rewritten Section VII-D2, and added a new table (Table VIII) and a new figure (Fig. 11) to report the running time.
5) We have added a new subsection (Section VI-D3) and a new figure (Fig. 12) to analyze the margin of label distribution and the influence of ρ.

C2: In addition, the paper needs to seriously address the concern on the completeness of literature review of the related subjects and the concern on the suitability of the paper for the journal – has TNNLS ever published a single paper on the topic described in this manuscript before? If yes, then why there is very little mentioning of such work(s) throughout the entire manuscript? If no, then why this type of topic would be of interest to the readers of the journal? This further creates the concern on the suitability of the paper for the journal.

R2: We appreciate these valuable comments. We have reviewed more TNNLS papers related to this study. The number of related TNNLS papers has been increased from three to seven. The referred TNNLS papers include:

1) [16] and [27] applied LDL methods to classification problems
2) [22] proposed an LDL methods
3) [35] and [36] applied large margin to multi-dimensional classification and partial label learning
4) [50] is an efficient $k$nn method, which learns different k's for different instances
5) [39] helps reduce the computational bottleneck of our methods, as suggested by Reviewer 1

Next, our responses to the detailed comments of the reviewers are as follows.

Reviewer 1:

In this paper, the authors have proposed two novel LDL methods, which are specially designed for classification. The proposed methods learn weight vectors for the kNN algorithm to learn label distribution and implement large margin to address the objective inconsistency. In general, this paper is well written and easy to follow. **I would like to accept this paper if my following concerns are carefully addressed:**

C1: The authors need to emphasize their contributions/novelties in the revision. In the current version, the authors did not discuss their contributions in detail.

R1: We appreciate this valuable comment. We further highlight our contributions as follows:

1) We have rewritten the sixth paragraph of Section I to better highlight our contributions. Our contributions mainly lie in that we propose two novel LDL methods for classification which learn weight vectors in the kNN algorithm to learn label distribution and implement large margin.
2) We have added more content in the second paragraph of Section II-A and rewritten the second paragraph of Section II-C to present the differences between our paper and related works to show our contributions.

C2: The proposed algorithm still can be improved if the ideas in the following papers are explored, i.e., "Compound Rank-k Projections for Bilinear Analysis" and "Self-weighted Robust LDA for Multiclass Classification with Edge Classes". The authors are encouraged to discuss them in the revision.

R2: Thanks for the suggestion. We have added more content in the second paragraph of **Section IV-C2** to cite the suggested papers and discuss how to improve our manuscript. We can use the suggested LDA methods to learn low-dimensional and discriminant features from the original ones, which would speed up the search process and reduce the computational complexity of the proposed methods.

C3: The authors should carefully proofread this paper and correct all the typos in the revision. In the current version, there are still some typos/grammar errors.

R3: We have carefully proofread the manuscript and tried our best to correct typos and errors.

C4: Could the authors report the running time of the proposed algorithm? In this way, we can justify whether this algorithm can be applied to large-scale dataset.

R4: Thanks for the valuable comment. We have rewritten Section VII-D2 "**Running Time**" as follows:

1) We have added a new table (Table VIII) to report the running time on three largest datasets. According to Table VIII, our methods need less running time than the comparing approaches.
2) We have compared the running time of our methods with brute-force searching against randomized kd-tree. We have added a new figure (Fig. 11) to report the results. The randomized kd-tree can reduce the quadratic running time to almost linear. Our methods apply to large-scale datasets with approximate nearest neighbor searching, such as the randomized kd-tree.

TABLE VIII
RUNNING TIME (S) OF FOUR METHODS ON THREE LARGE DATASETS.

Algorithm	Movie			Twitter_ldl			Flickr_ldl		
	train	test	total	train	test	total	train	test	total
RWLM-LDL	11.07	0.02	11.09	6.81	0.00	6.81	8.04	0.00	8.04
LDLM	6.75	0.01	6.76	3.41	0.00	3.42	3.74	0.00	3.74
LWkNN-LDL	0.72	0.51	1.22	0.89	0.38	1.27	1.02	0.47	1.49
LDkNN-LDL	0.74	0.50	1.24	0.91	0.38	1.29	1.08	0.48	1.56

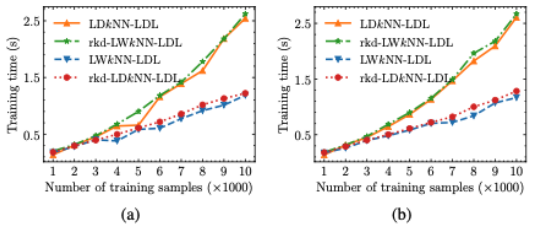


Fig. 11. Training time on (a) Flickr_ldl and (b) Twitter_ldl with an increasing number of training samples.

This article has been accepted for publication in IEEE Transactions on Neural Networks and Learning Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNNLS.2023.3297261

Reviewer 2:

The main contribution of this paper is to find the challenge of objective inconsistency between LDL and classification. To address the problem, the authors propose the LWkNN-LDL and LDkNN-LDL methods. First, they propose the LWkNN-LDL method. It learns a weight vector for the kNN algorithm to learn label distribution and implement a large margin to address the objective inconsistency. Second, they put forward the LDkNN-LDL approach that learns distance-dependent weight vectors to consider the difference in the neighborhoods of different instances. **The following suggestions may help improve this paper:**

C1: I think "LWkNN-LDL" is a short name for "Large Margin Weighted kNN LDL", then it should be defined first before authors use the "LWkNN-LDL" terminology in the Abstract section. (Also 'LDkNN-LDL')

R1: Thanks for pointing out this mistake. We have corrected it in the Abstract section and given the full names of the abbreviations for LWkNN-LDL and LDkNN-LDL.

C2: In order to study the generalization performance of the proposed model, a set of comparative experiments can be added at the end of the section 'VI B Classification Results and Discussion' to appropriately reduce the number of training samples and increase the number of test samples (70% and 30% for instance).

R2: We sincerely appreciate this constructive suggestion. We have conducted more experiments to compare LWkNN-LDL and LDkNN-LDL with RWLM-LDL and LDLM on eight datasets for ten times random data partitions (70% for training and 30% for testing) and added a new paragraph (the last second paragraph) to analyze the comparison results in **Section VI-B**.

We have added a new figure (Fig. 5) to report the comparison results in terms of 0/1 loss. From Fig. 5, LW k NN-LDL and LD k NN-LDL outperform RWLM-LDL and LDLM by a margin, which justifies the advantages of LWkNN-LDL and LDkNN-LDL for classification.

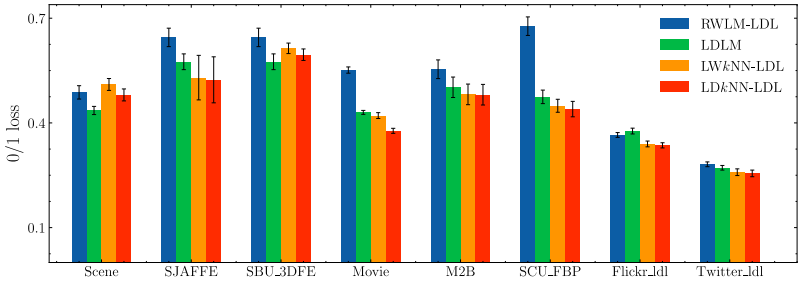


Fig.5 Further comparison with reduced training samples (70% for training and 30% for testing).

C3: Sometimes the authors should pay more attention to the situations where the proposed method works poorly and try to explain. For example, the WkNN and DkNN can not work as well as LDM when facing the LDL task because they do not consider label correlation. What about other experiments?

R3: We appreciate this valuable comment. We have added more content and analysis as follows:

1) We have added a new paragraph in Section VI-B and more content in the first paragraph of Section VI-C1. We have highlighted that LDL-LDM is a general LDL method, and our methods are designed for classification. As a result, LDL-LDM wins for the LDL experiments but our methods win for the classification experiments.
2) We have added a new subsection (Section VI-D3) to analyze when does our methods perform well. We have found that our methods perform well on datasets with large label distribution margin.

C4: (Minor) In Fig 6 and 7, the Y-axis is unclear.

R4: Thanks for pointing out this mistake. We have redrawn Figs. 7 and 8 (Figs. 6 and 7 change to 7 and 8) and annotated the label of Y-axis. Besides, we have added more content in the titles.

Reviewer 3:

The manuscript presented two new models in the category of Specialized Algorithm (SA), aiming to circumvent some previously detected misbehavior: (i) assume a label distribution; (ii) diminish the inconsistency between the objectives of LDL and classification.

C1: pg. 2 (line 34, 1º column) -> the characters of TRaditional word should be corrected.

R1: We appreciate the careful comment. We have checked it and found that this is due to the template of TNNLS. The code "\IEEEPARstart{T}raditional" gives "TRaditional" in the generated PDF.

C2: pg. 4 (line 54, 1º column) -> the authors claim: "In particular, our work differs from [36] in two aspects. First, [36] learns a metric for the kNN algorithm by large margin, while our work borrows large margin to learn weights for the kNN algorithm. Second, [36] only applies to SLL and does not suit LDL, but our methods are specially designed for LDL." **The reviewer thinks that the first aspect should be better clarified, i.e., the word borrows do not indicate the real difference between the models.**

R2: We appreciate this suggestion. We have rewritten the second paragraph of **Section I-C** to better justify the differences. We have highlighted that [36] learned a metric such that k-nearest neighbors have the same class and are separated from the samples with different classes by a margin. In comparison with [36], our work proposes to learn vector weights in kNN algorithm to learn label distribution and encourage large margin to address the objective inconsistency.

C3: pg. 4 (line 49, 2º column) -> as described in the text: "The LDL classifier f consider the label with the highest predicted label description degree as the predicted label".

Evaluating the adopted strategy pg. 5 (Eq. 8 and Eq. 11) -> The idea used to minimize objective inconsistency aims to guarantee a previously defined gap among the p values. **The reviewer has the following question:** an additional term was added to the objective function in both developed models to increase the margin. **Does this scheme work only if an inversion in the p vector order occur in relation to a model without this term?**

R3: Thanks for this comment. We have added more content in the last second paragraph of **Section IV-A** to explain that. A penalty would be added only if an inversion of order occurs in the predicted label distribution. Therefore, the models with large margin can avoid the occurrence of such inversion, which leads to better classification performance than the models without large margin.

C4: pg. 6 (line 18, 2º column) -> The section Optimization Method is very important. Usually, manuscript that needs optimization process neglects details of this phase.

pg. 7 -> Besides, the proposed methods are enhanced with theoretical developments. That is important. In pg. 7: The authors described: "We finish the proof of Eq. (14) by applying the Markov's inequality to the above inequality. Additionally, combining Eqs. (4) and (14), we complete the proof for Eq. (15)". It was interesting that the authors complete the description with the additional mathematical steps commented.

pg. 8 (line 20, 1º column) -> The real-world datasets adopted for the experiments are well representative and adequate. The simulations and comparisons in the first phase of Experiments section were very well guided, indicating directly the relevance and efficiency of the proposed methods in relation to relevant LDL methods.

However, the reviewer claims additional experiments in relation to subsection pg. 9 (line 50, 2º column) -> 2) Usefulness of Large Margin for Classification. **The results depicted in Fig 8 (pg. 12) did not indicate the relevance or importance to enforce large margin in the model**. The comparison between LWkNN-LDL (LDkNN-LDL) with λ_2 different to zero (large margin active) and equal to zero (large margin inactive) did not indicate the effectiveness of the term associated to the enforced margin gap.

This article has been accepted for publication in IEEE Transactions on Neural Networks and Learning Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNNLS.2023.3297261

R4: Thanks for this valuable comment. We have rewritten Section VI-C2 to better justify the advantages of large margin:

1) The differences are small on the first nine datasets because these datasets have small label distribution margins, as analyzed in **R5** below. So, we have redrawn Fig. 9 (Figs. 8 changes to 9 in the revision) to report the comparison results on the last eight datasets (from SJAFFE to Flickr_ldl) to better present the difference of performance.
2) To investigate whether large margin helps improve classification performance, we have conducted the pairwise *t*-tests by comparing LWkNN-LDL against WkNN-LDL and LDkNN-LDL against DkNN-LDL. We have added a new table (Table VII) to report the counts of win/tie/loss.

Fig. 9 shows that LWkNN-LDL and LDkNN-LDL achieve better performance than WkNN-LDL and DkNN-LDL. In particular, Table VII justifies that LWkNN-LDL and LDkNN-LDL statistically outperform WkNN-LDL and DkNN-LDL (67 wins out of 68 tests). Thus, the results indicate the advantages of large margin.

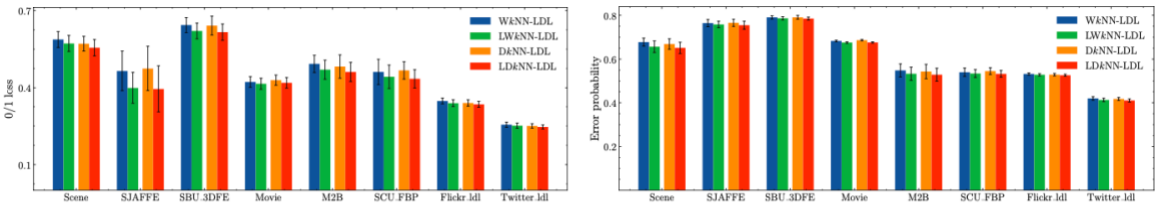


Fig. 9. Detailed results of WkNN-LDL, LWkNN-LDL, DkNN-LDL, and LDkNN-LDL in terms of (a) 0/1 loss and (b) error probability.

TABLE VII
EXPERIMENTAL RESULTS (WIN/TIE/LOSE COUNTS) OF THE PAIRWISE *t*-TESTS AT A CONFIDENCE LEVEL OF 0.05.

Metric	LWkNN-LDL *against*	LDkNN-LDL *against*
	WkNN-LDL	DkNN-LDL
0/1 loss	**17/0/0**	**17/0/0**
Error prob.	**16/1/0**	**17/0/0**

C5: The authors only used the value of $\rho = 0.1$ (pg. 9 (line 13, 1º column). This parameter seems to be very relevant to obtain adequate margin gap. **Its performance and influence should be more investigated.**

Finally, the reviewer evaluates that the idea based on large margin proposed by the authors **needs more analysis since for the ρ definition it is necessary to have a previous knowledge of the margin related to the dataset evaluated.** To define a value (as ρ = 0.1) could be inefficiency if the margin gap assumes around this value naturally. In this way, the reviewer proposes to the authors to add in Conclusion section some comments about this issue.

R5: We appreciate the valuable comments. We have added a new subsection (Section VI-D3) to analyze the margin of the datasets and added a new figure (Fig. 12) to investigate the influence of ρ.

We have defined the margin of the given label distribution. The margin of the given label distribution of $\boldsymbol{x}_i$ is defined by

$$\rho_i = d_{\boldsymbol{x}_i}^{y_{\ell_i}} - \max_{j \neq \ell_i} d_{\boldsymbol{x}_i}^{y_j}$$

which equals the difference between the highest and the second-highest label description degrees. We have added a new column in Table II to show the mean margin of each dataset. The last eight datasets have larger margins than the first nine ones, and LWkNN-LDL and LDkNN-LDL work more efficiently on the last eight datasets. That is, our methods tend to have better performance on datasets with large label distribution margins. Besides, we

This article has been accepted for publication in IEEE Transactions on Neural Networks and Learning Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNNLS.2023.3297261

have added more content in the second paragraph of the Conclusion section to discuss how to set ρ according to the margin of each dataset. We can set ρ to the mean margin and use different ρ's on different datasets. Also, we can set ρ to the margin of the given label distribution and use different ρ's for different training examples.

TABLE II
CHARACTERISTICS OF THE EXPERIMENTAL DATASETS.

ID	Dataset	*#Examples*	*#Features*	*#Labels*	avg. margin
1	Alpha	2,465	24	18	0.004
2	Cdc	2,465	24	15	0.005
3	Cold	2,465	24	4	0.034
4	Diau	2,465	24	7	0.014
5	Dtt	2,465	24	4	0.022
6	Elu	2,465	24	14	0.004
7	Heat	2,465	24	6	0.021
8	Spo	2,465	24	6	0.031
9	Spo5	2,465	24	3	0.083
10	SJAFFE	213	243	6	0.257
11	SBU_3DFE	2,500	243	6	0.090
12	Scene	2,000	294	9	0.131
13	Movie	7,755	1,869	5	0.104
14	M2B	1,240	250	5	0.513
15	SCUT_FBP	1,500	300	5	0.291
16	Twitter_ldl	10,040	200	8	0.369
17	Flickr_ldl	11,150	200	8	0.490

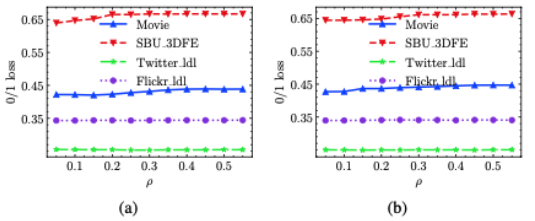


Fig. 12. Performance of (a) LWkNN-LDL and (b) LDkNN-LDL with ρ varying from 0.05 to 0.55 with a step of 0.05.

To investigate the influence of ρ, we have conducted extra experiments to run LWkNN-LDL and LDkNN-LDL with ρ varying from 0.05 to 0.55. We have added Fig. 12 to report their performance w.r.t. ρ. According to Fig. 12, we have founded:

1) Our methods are sensitive to ρ on datasets with small label distribution margins, like SBU 3DFE and Movie, where small values are preferred because large ones would far exceed the true margin of the given label distribution.
2) Our methods are robust to ρ on datasets with large label distribution margins, like Twitter ldl and Flickr ldl, since large values are tolerant by the true margins of the given label distribution.

Generally, $\rho = 0.1$ leads to satisfying performance for most of the datasets. So, we set $\rho = 0.1$ in the experiments. More efficient ways of setting ρ are added as future work.

Large Margin Weighted k-Nearest Neighbors Label Distribution Learning for Classification

Jing Wang and Xin Geng, *Senior Member, IEEE*

***Abstract*—Label Distribution Learning (LDL) helps solve label ambiguity and has found wide applications. However, it may suffer from the challenge of objective inconsistency when adopted to classification problems because the learning objective of LDL is inconsistent with that of classification. Some LDL algorithms have been proposed to solve this issue, but they presume that label distribution can be represented by the maximum entropy model, which may not hold in many real-world problems. In this paper, we design two novel LDL methods based on the k-Nearest Neighbors (kNN) approach without assuming any form of label distribution. First, we propose the Large margin Weighted kNN LDL (LW-kNNLDL). It learns a weight vector for the kNN algorithm to learn label distribution and implement large margin to address the objective inconsistency. Second, we put forward the Large margin Distance-weighted kNN LDL (LDkNN-LDL) that learns distance-dependent weight vectors to consider the difference in the neighborhoods of different instances. Theoretical results show that our methods can learn any general-form label distribution. Moreover, extensive experimental studies validate that our methods significantly outperform the state-of-the-art LDL approaches.**

***Index Terms*—Label distribution learning (LDL), k-nearest neighbors (kNN), weighted kNN, large margin, classification, generalization**

I. Introduction

Traditional supervised learning paradigms, such as Single-Label Learning (SLL) and Multi-Label Learning (MLL), assume that labels are certainly relevant or irrelevant to instances. However, in many real-world problems, the relation between instances and labels often contains somewhat uncertainty [1], known as the label ambiguity [2] in the field of machine learning. For instance, Fig. 1 shows an example image from the JAFFE [3] database with the ground-truth label "ANG". The image blends multiple emotions with different relevance, which include not only "ANG" but also some other emotions with less relevance. However, SLL and MLL use 0 and 1 to represent the relation between instances and labels, which ignores label ambiguity.

Geng [4] proposed a novel learning paradigm called Label Distribution Learning (LDL) to address label ambiguity. LDL uses, instead of binary values as by SLL and MLL, real-values to model the relation between instances and labels. It assigns label distribution to each instance whose elements are defined as the label description degrees and represent the relevance degrees of labels. For example, the image in Fig. 1 is annotated with label distribution[1] $[0.11, 0.19, 0.09, 0.32, 0.29]^\top$, in which the ground-truth label "ANG" has the largest label description degree 0.32, and the other labels have fewer ones. LDL can explicitly express the different relevance of labels, which is more suit for such applications.

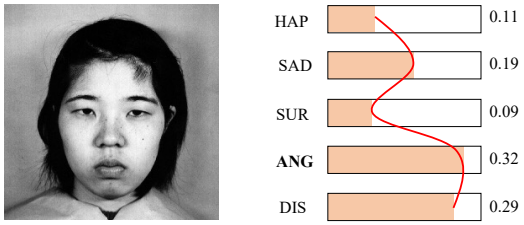


Fig. 1. An example image from the JAFFE [3] database with the ground-truth label "ANG". The mean ratings are normalized into label distribution.

LDL aims to learn the label distribution of training instances and predict on unknown ones. It also applies to classification problems as follows. First, an LDL function is learned from the training set with label distribution. Then, the learned LDL function is used for classification. For an unknown instance, the label with the highest predicted label description degree by the learned LDL function is treated as the predicted label. LDL has already found extensive applications in various real-world problems, such as facial age estimation [5]–[7], emotion recognition [8]–[10], head-pose estimation [11], sentiment analysis [12], [13], beauty perception [14], [15], and noisy label learning [16]. It has shown advantages over the traditional SLL in these tasks.

However, LDL may suffer from the challenge of objective inconsistency [17] when applied to classification problems. Specifically, LDL aims to learn the whole label distribution (e.g., the label distribution in Fig. 1), but the goal of classification is to learn the optimal label with the highest label description degree (e.g., "ANG" in Fig. 1) — the objective of LDL is inconsistent with that of classification. Fig. 2 presents an example to demonstrate that. In Fig. 2(a), the predicted label distribution has an L_1-norm loss of 0.20, but the predicted label (i.e., y_5) is different from the optimal label (i.e., y_4). As for Fig. 2(b), it achieves an L_1-norm loss of 0.30, and its predicted label (i.e., y_4) equals the optimal label. To summarize,

Manuscript received April 19, 2005; revised August 26, 2015. This work was supported in part by the National Key Research and Development Plan of China under Grant 2018AAA0100104, in part by the National Natural Science Foundation of China under Grant 62076063, and in part by the Fundamental Research Funds for the Central Universities under Grant 2242021k30056. (*Corresponding author: Xin Geng.*)

The authors are with the School of Computer Science and Engineering, Southeast University, Nanjing 211189, China, and also with the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing 211189, China (email: wangjing91@seu.edu.cn; xgeng@seu.edu.cn).

[1]The label distribution is obtained by normalizing the mean ratings.

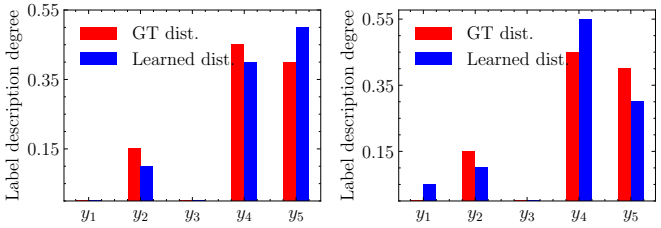


Fig. 2. An illustration of the objective inconsistency, where the red and blue bars denote the ground-truth and learned label distribution, respectively. (a) has an L_1-norm loss of 0.20 and a classification loss of 1. (b) achieves an L_1-norm loss of 0.30 and a classification loss of 0. (a) is better than (b) in terms of LDL but worse than (b) from the perspective of classification.

Fig. 2(a) is superior to 2(b) in terms of LDL but inferior to 2(b) from the perspective of classification. The reason is that LDL may ignore the optimal label for the sake of learning the whole label distribution [17]. As a result, one may not expect satisfying classification performance even if label distribution is well learned. Existing general-purpose LDL methods aiming to learn label distribution ignore the objective inconsistency and may have sub-optimal classification performance.

We have proposed several specially designed LDL methods for classification [17]–[19], which can address the objective inconsistency and have better classification performance. Like many general-purpose LDL methods [4], [5], [20]–[22], they all assume that label distribution can be represented by the maximum entropy model [23] and employ it to learn label distribution. However, this assumption may not hold sometimes, especially for mixture distribution [24], which may reduce the performance of these specialized methods.

In this paper, we propose two novel LDL methods for classification. Without presuming any form of label distribution, our methods employ the k-Nearest Neighbors (kNN) algorithm to learn label distribution, which differ from those relying on the maximum entropy model. First, we put forward the Large margin Weighted kNN Label Distribution Learning (LWkNN-LDL). It learns a weight vector for the kNN algorithm to learn label distribution and implement large margin to address the objective inconsistency. However, LWkNN-LDL learns a fixed weight vector for all instances and ignores the difference in the neighborhoods of different instances. To address that, we learn distance-dependent weight vectors and propose the Large margin Distance-weighted kNN Label Distribution Learning (LDkNN-LDL). Different from the kNN algorithm, our methods need training phases to learn the weight vectors, which not only help learn label distribution but also solve the objective inconsistency by implementing large margin.

In the theoretical results, we prove $\mathcal{O}[(k/n)^{1/q+1}]$ bounds (n and q are the sizes of the training set and feature dimensions, respectively) on the expected L_1-norm loss and the classification error of the proposed methods, which approach 0 as n goes to infinity. The theoretical results suggest that the proposed methods can learn any general-form label distribution, and their classification error may approach the Bayes error [25], justifying their theoretical advantages. Moreover, in the experiments, the proposed methods achieve significantly better classification performance than several state-of-the-art general-purpose and specially designed LDL methods, which validates their superiority for classification.

The major contributions of this paper include:

1) We designed two novel LDL approaches based on the kNN algorithm. They learn weight vectors for the kNN algorithm to learn label distribution and implement large margin to solve the objective inconsistency.
2) We analyze the generalization of the proposed methods. The theoretical results disclose that they can learn any general form of label distribution under a mild assumption of Lipschitzness.
3) We conduct extensive experiments to validate the proposed methods. The experimental results show that they achieve significantly superior classification performance than the state-of-the-art LDL methods.

We organize the rest of the paper as follows. Section II reviews related works. Section III introduces some background. Next, Section IV presents the proposed methods. Sections V and VI report the theoretical and experimental results, respectively. Finally, Section VII concludes the paper.

II. Related Work

A. Label Distribution Learning

Geng et al. [5] first introduced label distribution to facial age estimation. They used label distribution to model the slow and smooth change of facial appearance [5] in the aging process, and proposed two algorithms to learn from such distribution. Later, Geng formalized LDL as a new learning paradigm [4] that differs from traditional learning paradigms in learning label distribution. He also suggested three strategies to design new LDL algorithms, including Problem Transformation (PT), Algorithm Adaptation (AA), and Specialized Algorithm (SA), and put forward several representative LDL baselines, such as PT-SVM, AA-kNN, and SA-BFGS [4].

Since then, researchers have proposed many LDL methods. Geng and Hou [26] viewed LDL as a regression problem and proposed LDL-Support Vector Regression (SVR). It applies the multivariate SVR to learn label distribution. Shen et al. [24] employed the differentiable forest to learn label distribution and designed LDLFs. It can be combined with any representative learning models (e.g., convolutional neural network) in an end-to-end way. Jia et al. [21] exploited local label correlation in LDL and put forward LDL-SCL. It encodes label correlation as additional features and jointly learns label distribution and the encoding of label correlation. In addition, we considered both global and local label correlations in LDL and proposed LDL-LDM [22] that exploits label correlation by learning label distribution manifold.

The preceding works propose several general-purpose LDL methods to learn label distribution. In this study, we design two specialized LDL methods for classification, which help solve the objective inconsistency.

B. LDL for Classification

LDL has seen extensive applications in various classification problems. Geng et al. [11] used Multivariate Label Distribution

(MLD) to annotate images for head-pose estimation. MLD covers the ground-truth pose and neighborhood poses. They learned mapping from images to MLD. For an unknown image, they regarded the pose with the highest label description degree in the predicted MLD as the prediction [11]. In age estimation, Shen et al. [7] learned an LDL model from the facial images described by (age) label distribution. Similarly, they treated the age label having the highest predicted label description degree as the predicted age. Shu et al. [10] formalized the problem of emotion recognition as emotion distribution learning and annotated each image by emotion distribution. They learned an LDL model from such emotion distribution and took the dominant emotion in the predicted distribution as the prediction. Jiang et al. [27] employed label distribution to represent the noisy label set of each instance from crowdsourcing and propagated it to neighbors. Then, they regarded the dominant label in the estimated label distribution as the integration label. Moreover, Xu et al. [16] applied LDL to noisy label learning, Fan et al. [15] introduced LDL to facial beauty perception, and Yang et al. [13] employed LDL to image sentiment analysis. However, these works apply general-purpose LDL methods to classification problems and ignore the inconsistency between the objectives of LDL and classification.

Our previous work [17] studied LDL for classification and found that the expected loss of LDL bounds the classification error. Besides, we have put forward several specially designed LDL methods for classification. Our earlier research [17] introduced a new re-weighing scheme and large margin to LDL. We established the label distribution margin theory [18] and designed a new LDL method for classification. Moreover, we viewed label distribution as a combination of the optimal label and rest label description degrees and learned both of them [19]. However, these methods assume that label distribution can be represented by the maximum entropy model, which may not hold [24]. Instead, this study applies the kNN algorithm to LDL, without assuming any form of label distribution.

C. Weighted kNN and Large Margin

The kNN method makes estimations for unknown instances based on the observations of their k-nearest neighbors [28], [29]. Dudani [30] proposed a distance-weighted kNN method. He suggested closer neighbors should have a greater influence than distant ones. To achieve that, Dudani weighted k-nearest neighbors by the inverse of their distances [30]. However, the weights are obtained heuristically, which only depend on feature space and are independent of label space. Our work learns distance-dependent weights from training data, which depend on both feature space and label distribution and are more data-dependent.

Large margin [31] has motivated many learning approaches. Cortes and Vapnik [32] put forward the Support Vector Machine (SVM) to maximize the margin of training data. Wu and Zhou [33] applied large margin to MLL and maximized label- and instance-wise margins. Jia and Zhang [34] adapted large margin to multi-dimensional classification and maximized the sum of margins across all dimensions. Chai et al. [35] extended multi-class SVM to partial-label learning. Weinberger and Saul [36] learned a metric that k-nearest neighbors have the same class and are separated from the samples with different classes by a margin. In this study, we combine the kNN method and large margin to LDL for classification. We learn vector weights in the kNN method to learn label distribution and implement large margin.

TABLE I
DEFINITIONS OF THE MAINLY USED NOTATIONS

Symbol	Definition
$\mathcal{X} \subseteq \mathbb{R}^q$ and $\mathcal{Y}$	q-D feature space and label Space
$\boldsymbol{x}_i$ and D_i	The ith instance and label distribution
$\boldsymbol{x}_{i,j}$	The jth nearest neighbor of $\boldsymbol{x}_i$
$D_{i,j}$	The label distribution of $\boldsymbol{x}_{i,j}$
$\boldsymbol{D}_i$	The kNN label distribution matrix of $\boldsymbol{x}_i$
$\boldsymbol{d}_i$	The kNN distance vector of $\boldsymbol{x}_i$
ℓ_i	The index of the optimal label for $\boldsymbol{x}_i$
$[\boldsymbol{I}]_{i,l}$ and $[\boldsymbol{I}]_l$	The (i,l) element and lth row vector of $\boldsymbol{I}$
$\|\boldsymbol{W}\|_{\mathrm{F}}$	The Frobenius norm of a matrix $\boldsymbol{W}$
$\|\boldsymbol{w}\|_p$	The L_p-norm of a vector $\boldsymbol{w}$
$\mathbb{I}(\cdot)$	The indicator function
$\phi(\cdot)$	Element-wise sign operator
$[m]$	The set $\{1, 2, \cdots, m\}$

III. BACKGROUND

We start with the notations. Let $\mathcal{X} \subseteq \mathbb{R}^q$ be the q-D feature space and $\mathcal{Y} = \{y_1, y_2, \cdots, y_m\}$ be the label space with m candidate labels. In the settings of LDL, each instance $\boldsymbol{x}$ has label distribution denoted by $D = [d_{\boldsymbol{x}}^{y_1}, d_{\boldsymbol{x}}^{y_2}, \cdots, d_{\boldsymbol{x}}^{y_m}]^\top$, where $d_{\boldsymbol{x}}^{y_j}$ is the label description degree and satisfies the probability simplex constraint, i.e., $d_{\boldsymbol{x}}^{y_l} \geq 0$ and $\sum_l d_{\boldsymbol{x}}^{y_l} = 1$ [4]. Let $S = \{(\boldsymbol{x}_1, D_1), (\boldsymbol{x}_2, D_2), \cdots, (\boldsymbol{x}_n, D_n)\}$ stand for a training set, where $\boldsymbol{x}_i$ is the ith training instance, and $D_i = [d_{\boldsymbol{x}_i}^{y_1}, d_{\boldsymbol{x}_i}^{y_2}, \cdots, d_{\boldsymbol{x}_i}^{y_m}]^\top$ is its label distribution. Table I summarizes the mainly used notations.

A. LDL and Classification

LDL aims to learn mapping $p : \mathcal{X} \to \mathbb{R}^m$ from S. Given a loss function $\ell : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}^+$ (e.g., the Kullback–Leibler (KL) divergence [4]), LDL can be formalized by

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \ell(D_i, p(\boldsymbol{x}_i; \boldsymbol{\theta})), \tag{1}$$

where p is typically a parametric model. As discussed above, LDL can be applied to classification problems. First, an LDL function p is learned from S as Eq. (1). Second, an LDL classification function f is derived from p by

$$f(\boldsymbol{x}) = \arg\max_{\bar{y} \in \mathcal{Y}} p_{\boldsymbol{x}}^{\bar{y}}, \tag{2}$$

where $p_{\boldsymbol{x}}^{\bar{y}}$ is the predicted label description degree of $\bar{y}$. The LDL classifier f regards the label with the highest predicted label description degree as the predicted label.

We studied the generalization of LDL classification in [17]. Let y be the random label variable. Define the error of f by

$$L(f) = \mathbb{E}_{\boldsymbol{x},y}\left[\mathbb{I}(f(\boldsymbol{x}) \neq y)\right], \tag{3}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Let L^* denote the Bayes error [25] (i.e., the possible minimal error). Suppose the given label distribution is the conditional probability distribution. Our previous study [17] has proved the following theorem.

***Theorem** 1 (Wang et al. [17]):* Let p be a learned LDL function and f be the LDL classification function as defined in Eq. (2). Then, the error of f satisfies

$$L(f) - L^* \leq \mathbb{E}_{\boldsymbol{x}}\left[\sum_{l=1}^{m} |d_{\boldsymbol{x}}^{y_l} - p_{\boldsymbol{x}}^{y_l}|\right]. \tag{4}$$

In Eq. (4), the left-hand side is the difference between the error of f and the Bayes error, and the right-hand side is the expected L_1-norm loss of p. Theorem 1 says that the expected L_1-norm loss of p bounds the classification error of f. That is, LDL dominates classification [17].

B. The AA-kNN Algorithm

AA-kNN adapts the kNN algorithm to LDL [4]. For a test instance $\boldsymbol{x}$, it calculates the mean of the label distribution of its k nearest neighbors as the prediction, i.e.,

$$p(\boldsymbol{x}) = \frac{1}{k}\sum_{l \in N_k(\boldsymbol{x})} D_l, \tag{5}$$

where $N_k(\boldsymbol{x})$ is the index set of the k-nearest neighbors of $\boldsymbol{x}$. AA-kNN does not presume any form of label distribution and can learn any general-form ones [37]. But it is a general-purpose LDL method and suffers from the objective inconsistency when applied to classification problems.

IV. The Proposed Approaches

This section presents the proposed methods. We first introduce LWkNN-LDL and LDkNN-LDL and then elaborate on the optimization method.

A. The LWkNN-LDL Approach

This subsection presents LWkNN-LDL. We start with the following two observations:

- From Theorem 1, to minimize the classification error of an LDL classifier, it suffices to minimize the expected L_1-norm loss in LDL.
- AA-kNN applies the kNN algorithm to LDL. It can learn any general-form label distribution [37], which shows the advantage of the kNN algorithm.

To summarize, we can expect to optimize the classification error by minimizing the L_1-norm loss in LDL and learn any general-form label distribution by the kNN model. However, these two cannot be simply combined, as the kNN algorithm does not involve a training phase to minimize the L_1-norm loss. Our basic idea is to use the weighted kNN method and learn a weight vector for the kNN algorithm to directly minimize the L_1-norm loss of LDL.

From Eq. (5), AA-kNN can also be viewed as a weighted kNN method with a fixed weight vector of $[1/k, \cdots, 1/k]$ for all instances. Inspired by that, we propose to learn a fixed weight vector for all instances. Concretely, for a training instance $\boldsymbol{x}_i$, let $\boldsymbol{x}_{i,j}$ stand for its jth nearest neighbor in the training set (excluding itself) and $D_{i,j}$ be the label distribution of $\boldsymbol{x}_{i,j}$. Define the kNN label distribution matrix of $\boldsymbol{x}_i$ by $\boldsymbol{D}_i = [D_{i,1}, D_{i,2}, \cdots, D_{i,k}]$, where the jth column is the label distribution of the jth nearest neighbors. Let $\boldsymbol{w}$ denote the weight vector to be learned. Then, the predicted label distribution of $\boldsymbol{x}_i$ equals

$$p(\boldsymbol{x}_i) = \sum_l w_l \cdot D_{i,l} = \boldsymbol{D}_i\boldsymbol{w}, \tag{6}$$

where w_l is the lth element of $\boldsymbol{w}$. That is, for any instance, the lth nearest neighbor is always weighted with w_l. Applying the L_1-norm loss as the learning metric, we design the following optimization objective:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{n} \|D_i - \boldsymbol{D}_i\boldsymbol{w}\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{w}\|_2^2, \tag{7}$$

where λ_1 is a regularization parameter. AA-kNN is a special case of model (7) by setting $w_l = 1/k$ instead of learning it from the training set.

Model (7) implicitly minimizes classification error by optimizing the L_1-norm loss. However, it is still a general-purpose LDL method and suffers from the objective inconsistency. To address that, we apply the margin theory [32] to model (7) to improve its classification performance. Our basic idea is to learn a weight vector to implement large margin.

By introducing large margin, we expect the predicted label description degree of the optimal label to be larger than those of other labels by a margin. The optimal label would also have the highest degree in the predicted label distribution, incurring no classification loss. Specifically, for $\boldsymbol{x}_i$, define the index of the optimal label by

$$\ell_i = \arg\max_{j \in [m]} d_{\boldsymbol{x}_i}^{y_j}$$

and the optimal label by y_{ℓ_i}. We encourage the predicted label description degree of y_{ℓ_i} to be larger than that of y_l by at least a margin ρ for $l \neq \ell_i$. Adding large margin to model (7), we design the following objective function for LWkNN-LDL:

$$\begin{aligned} \min_{\boldsymbol{w}} & \sum_{i=1}^{n}\|D_i - \boldsymbol{D}_i\boldsymbol{w}\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{w}\|_2^2 + \lambda_2 \sum_{i=1}^{n}\sum_{l \neq \ell_i} \frac{\xi_{i,l}}{\rho} \\ \text{s.t. } & [\boldsymbol{D}_i]_{\ell_i}\boldsymbol{w} - [\boldsymbol{D}_i]_l\boldsymbol{w} \geq \rho - \xi_{i,l} \\ & \xi_{i,l} \geq 0, \text{for } i \in [n], l \in [m], \text{and } l \neq \ell_i, \end{aligned} \tag{8}$$

where $[\boldsymbol{D}_i]_l$ denotes the lth row vector of $\boldsymbol{D}_i$, $\xi_{i,l}$ is the slack variable, and λ_2 is a trade-off parameter. The first constraint encourages the predicted label description degree of y_{ℓ_i} (i.e., $[\boldsymbol{D}_i]_{\ell_i}\boldsymbol{w}$) to be larger than that of y_l (i.e., $[\boldsymbol{D}_i]_l\boldsymbol{w}$) by a margin of ρ. If it holds, $\xi_{i,l}$ would equal 0, which incurs no loss in the objective function. Otherwise, $\xi_{i,l} > 0$, which adds a loss of $\xi_{i,l}/\rho$ to the objective function. Compared with Eqs. (7), (8) encourages satisfaction of the first constraint, which helps solve the objective inconsistency.

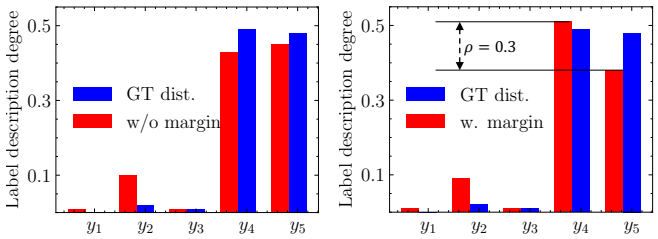


Fig. 3. Explaining the advantage of large margin by showing LWkNN-LDL without and with large margin in (a) and (b), respectively. (a) achieves a smaller L_1-norm loss of 0.18 but misses the optimal label. Although (b) has a higher L_1-norm loss of 0.20, its predicted label equals the optimal label because it encourages a margin of 0.3.

We give an example in Fig. 3 to illustrate the advantage of large margin. The optimal label in the ground-truth distribution is y_4. Fig. 3(a) shows that LWkNN-LDL without large margin achieves an L_1-norm loss of 0.18 and its predicted label is y_5. Fig. 3(b) shows that LWkNN-LDL with large margin has an L_1-norm loss of 0.20 and its predicted label is y_4. Fig. 3(a) achieves a smaller L_1-norm loss but misses the optimal label because it aims to learn label distribution. Fig. 3(b) encourages a margin of 0.3 between the optimal label and other labels in the predicted distribution. Its predicted label equals the optimal label even though it has a higher L_1-norm loss.

B. The LDkNN-LDL Approach

LWkNN-LDL learns a fixed weight vector for all instances, which ignores the difference in the neighborhoods of different instances. To see that, we show an example in Fig. 4(a). The learned weight vector equals $[0.4, 0.3, 0.3]^\top$, which weights the first, the second, and the third nearest neighbors with 0.4, 0.3, and 0.3, respectively. For $\boldsymbol{x}_l$, its three nearest neighbors have similar distances. For $\boldsymbol{x}_i$, the third nearest neighbor is far more distant from $\boldsymbol{x}_i$ than the other two. However, $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ share the same weight vector, which neglects the differences in their neighborhoods.

To solve the above problem, we can learn different weight vectors for different instances. Moreover, the example in Fig. 4(a) suggests that the optimal weight vector of each instance may depend on the distances of its k-nearest neighbors, which inspires us to learn distance-dependent weight vectors. Denote by $\mathrm{Dis}(\boldsymbol{x}_i, \boldsymbol{x}_l) = \|\boldsymbol{x}_i - \boldsymbol{x}_l\|_2$ the Euclidean distance from $\boldsymbol{x}_i$ to $\boldsymbol{x}_l$. Define the vector of distances from $\boldsymbol{x}_i$ to its k-nearest neighbors by

$$\boldsymbol{d}_i = [\mathrm{Dis}(\boldsymbol{x}_i, \boldsymbol{x}_{i,1}), \mathrm{Dis}(\boldsymbol{x}_i, \boldsymbol{x}_{i,2}), \cdots, \mathrm{Dis}(\boldsymbol{x}_i, \boldsymbol{x}_{i,k})]^\top,$$

whose the lth element is the distance from $\boldsymbol{x}_i$ to its lth nearest neighbor. We use a distance-dependent weight vector for $\boldsymbol{x}_i$, i.e., $\boldsymbol{w}_i = \boldsymbol{W}\boldsymbol{d}_i$, where $\boldsymbol{W} \in \mathbb{R}^{k\times k}$ is a parameter matrix. Then, the predicted label distribution for $\boldsymbol{x}_i$ equals

$$p(\boldsymbol{x}_i) = \sum_l w_{i,l} \cdot D_{i,l} = \boldsymbol{D}_i\boldsymbol{w}_i = \boldsymbol{D}_i\boldsymbol{W}\boldsymbol{d}_i \tag{9}$$

where $w_{i,l}$ is the lth element of $\boldsymbol{w}_i$. Using the L_1-norm loss, we design the following optimization objective:

$$\min_{\boldsymbol{W}} \sum_{i=1}^n \|D_i - \boldsymbol{D}_i\boldsymbol{W}\boldsymbol{d}_i\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{W}\|_{\mathrm{F}}^2, \tag{10}$$

where $\|\boldsymbol{W}\|_{\mathrm{F}}$ denotes the Frobenius norm of $\boldsymbol{W}$.

Fig. 4(b) explains the advantage of the distance-dependent weight vectors. Compared with Figs. 4(a) that learns a fixed weight vector of $[0.4, 0.3, 0.3]^\top$ for both $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$, 4(b) learns distance-dependent weight vectors of $[0.5, 0.4, 0.1]^\top$ and $[0.4, 0.3, 0.3]^\top$ for $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$, respectively, which adapt to their neighborhoods.

Similar to (7), model (10) suffers from the objective inconsistency. Likewise, we can add large margin to Eq. (10) and design the following objective function for LDkNN-LDL:

$$\begin{aligned}\min_{\boldsymbol{W}} \sum_{i=1}^n \|D_i - \boldsymbol{D}_i\boldsymbol{W}\boldsymbol{d}_i\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{W}\|_{\mathrm{F}}^2 + \lambda_2 \sum_{i=1}^n \sum_{l\neq\ell_i} \frac{\xi_{i,l}}{\rho} \\ \text{s.t. } [\boldsymbol{D}_i]_{\ell_i}\boldsymbol{W}\boldsymbol{d}_i - [\boldsymbol{D}_i]_l\boldsymbol{W}\boldsymbol{d}_i \geq \rho - \xi_{i,l} \\ \xi_{i,l} \geq 0, \text{for } i \in [n], l \in [m], \text{ and } l \neq \ell_i.\end{aligned} \tag{11}$$

The first constraint encourages the predicted label description degree of y_{ℓ_i} (i.e., $[\boldsymbol{D}_i]_{\ell_i}\boldsymbol{W}\boldsymbol{d}_i$) to be large than that of y_l ($[\boldsymbol{D}_i]_l\boldsymbol{W}\boldsymbol{d}_i$) by a margin ρ for $l \neq \ell_i$.

C. Optimization Method

Eqs. (8) and (11) are convex and have $\mathcal{O}(nm)$ constraints. Here, we apply the gradient descent method [38] to solve them. Algorithm 1 briefly summarizes LDkNN-LDL (LWkNN-LDL can be summarized in a similar way as Algorithm 1).

1) Parameter Update: To start, define the ρ-margin loss function by $\ell_\rho(x) = \max(0, 1 - x/\rho)$. We can re-cast Eq. (8) as the following unconstrained optimization problem:

$$\min_{\boldsymbol{w}} \sum_{i=1}^n \|D_i - \boldsymbol{D}_i\boldsymbol{w}\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{w}\|_2^2 + \lambda_2 \sum_{i=1}^n \sum_{l\neq\ell_i} \ell_\rho(\Delta_{i,l}),$$

where $\Delta_{i,l} = [\boldsymbol{D}_i]_{\ell_i}\boldsymbol{w} - [\boldsymbol{D}_i]_l\boldsymbol{w}$. For simplicity of notation, define the indicator matrix $\boldsymbol{I}$ by

$$[\boldsymbol{I}]_{i,l} = \mathbb{I}(\Delta_{i,l} < \rho) \text{ for } l \neq \ell_i, \text{ and } [\boldsymbol{I}]_{i,\ell_i} = -\sum_{l\neq\ell_i} [\boldsymbol{I}]_{i,l}.$$

The above unconstrained problem has the following gradient:

$$\nabla_{\boldsymbol{w}} = \sum_{i=1}^n \boldsymbol{D}_i^\top \phi(\boldsymbol{D}_i\boldsymbol{w} - D_i) + \lambda_1\boldsymbol{w} + \frac{\lambda_2}{\rho}\sum_{i=1}^n \boldsymbol{D}_i^\top [\boldsymbol{I}]_i^\top$$

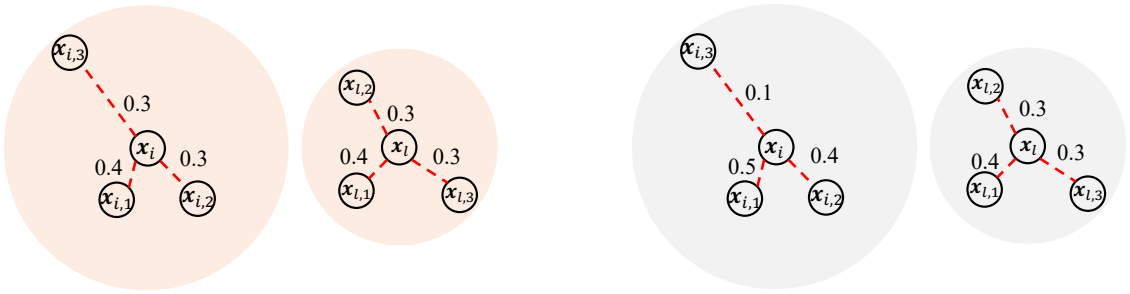
where $\phi(\cdot)$ is the element-wise sign operator, and $[\boldsymbol{I}]_i$ denotes the ith row vector of $\boldsymbol{I}$. We apply the steepest descent method [38] to optimize the unconstrained objective. In the tth step, $\boldsymbol{w}$ is updated by

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} - \eta_t \cdot \nabla_{\boldsymbol{w}_{t-1}} \tag{12}$$

where η_t is the step length obtained from the line search [38].

In addition, we can re-formulate Eq. (11) as the following unconstrained optimization problem:

$$\min_{\boldsymbol{w}} \sum_{i=1}^n \|D_i - \boldsymbol{D}_i\boldsymbol{W}\boldsymbol{d}_i\|_1 + \frac{\lambda_1}{2}\|\boldsymbol{W}\|_{\mathrm{F}}^2 + \lambda_2 \sum_{i,l\neq\ell_i} \ell_\rho(\Delta'_{i,l}),$$



(a) (b)

Fig. 4. An illustration of the difference between (a) learning a fixed weight vector and (b) learning distance-dependent weight vectors. The neighbors of $\boldsymbol{x}_l$ have similar distances to it. The third nearest neighbor of $\boldsymbol{x}_i$ is far more distant from it than other two. For (a), $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ share the same learned weight vector of $[0.4, 0.3, 0.3]^\top$, which ignores the differences of their neighborhoods. (b) learns distance-dependent weight vectors of $[0.5, 0.4, 0.1]^\top$ and $[0.4, 0.3, 0.3]^\top$ for $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$, respectively, which considers the difference of their neighborhoods.

where $\Delta'_{i,l} = [\boldsymbol{D}_i]_{\ell_i} \boldsymbol{W} \boldsymbol{d}_i - [\boldsymbol{D}_i]_l \boldsymbol{W} \boldsymbol{d}_i$. Define $\boldsymbol{I}'$ by

$$[\boldsymbol{I}']_{i,l} = \mathbb{I}(\Delta'_{i,l} < \rho) \text{ for } l \neq \ell_i, \text{ and } [\boldsymbol{I}']_{i,\ell_i} = -\sum_{l \neq \ell_i} [\boldsymbol{I}']_{i,l}.$$

Then, the gradient of the above unconstrained problem can be calculated as the following:

$$\nabla \boldsymbol{W} = \sum_{i=1}^{n} \boldsymbol{D}_i^\top \left(\phi(\boldsymbol{D}_i \boldsymbol{W} \boldsymbol{d}_i - D_i) + \frac{\lambda_2 [\boldsymbol{I}]_i^\top}{\rho} \right) \boldsymbol{d}_i^\top + \lambda_1 \boldsymbol{W}$$

Similarly, we apply the steepest descent method [38] and update $\boldsymbol{W}$ in the tth step by

$$\boldsymbol{W}_t = \boldsymbol{W}_{t-1} - \eta'_t \cdot \nabla \boldsymbol{W}_{t-1} \tag{13}$$

where η'_t is the step length by the line search [38].

Algorithm 1 summarizes LDkNN-LDL. Step 1 finds the k-nearest neighbors for each training instance to calculate the kNN label distribution matrix and distance vector. Then, steps 2 to 5 update the parameter $\boldsymbol{W}$. For the test instance $\boldsymbol{x}$, steps 6 and 7 search for its k-nearest neighbors and compute the kNN distance vector $\boldsymbol{d}$ and label distribution matrix $\boldsymbol{D}$. Step 8 computes the weight vector $\boldsymbol{w}$. Step 9 clips and normalizes $\boldsymbol{w}$ so that the predicted distribution will satisfy the probability simplex constraint. Finally, step 10 calculates and returns the predicted label distribution.

Algorithm 1 The LDkNN-LDL Approach

Input: training set S, and a test instance $\boldsymbol{x}$
Parameter: λ_1, λ_2, k, ρ, T

1: Calculate $\boldsymbol{D}_i$ and $\boldsymbol{d}_i$ for each training instance $\boldsymbol{x}_i$
2: Randomly initialize $\boldsymbol{W}$
3: **for** $t = 1$ to T **do**
4: Update $\boldsymbol{W}$ according to Eq. (13)
5: **end for**
6: Calculate the kNN distance vector $\boldsymbol{d}$ for $\boldsymbol{x}$
7: Calculate the kNN label distribution matrix $\boldsymbol{D}$ for $\boldsymbol{x}$
8: Compute the weight vector $\boldsymbol{w} = \boldsymbol{W}\boldsymbol{d}$
9: Clip $\boldsymbol{w}$ by $w_i = \max(0, w_i)$ and normalize $\boldsymbol{w}$
10: **return** $\boldsymbol{D}\boldsymbol{w}$

2) Computational Cost: Algorithm 1 has $\mathcal{O}(Tnmk^2 + n^2 q)$ computational complexity. First, we need $\mathcal{O}(n^2 q)$ complexity to find the k-nearest neighbors of all training instances. Second, in each iteration of parameter update, computing $\boldsymbol{D}_i \boldsymbol{W} \boldsymbol{d}_i$ for all $i \in [n]$ has $\mathcal{O}(nmk^2)$ complexity, the calculation of $\boldsymbol{I}'$ needs $\mathcal{O}(nm)$ complexity, and the computation of the gradient needs $\mathcal{O}(nkm + nk^2)$ complexity, which yields $\mathcal{O}(Tnmk^2)$ complexity for T iterations. Consequently, the total complexity for Algorithm 1 is $\mathcal{O}(n^2 q + Tnmk^2)$. Similarly, the total computational complexity of the LWkNN-LDL approach is $\mathcal{O}(Tnmk + n^2 q)$.

The computation bottleneck of Algorithm 1 lies in searching the k-nearest neighbors for all training instances. To address that, we can use the Linear Discriminant Analysis (LDA) [39], [40] to learn low-dimensional and discriminant features from the original ones, which would speed up the search process. Moreover, we can borrow approximate nearest neighbor search [41] instead of brute-force search. For example, the randomized kd-tree method [42] can reduce the computational cost of Algorithm 1 to $\mathcal{O}(kqn \log n + Tnmk^2)$.

V. Theoretical Results

In this section, we analyze the generalization of LWkNN-LDL and LDkNN-LDL. For the convenience of analysis, we assume $\mathcal{X} \subseteq [0,1]^q$ and the given label distribution satisfies α-Lipschitzness, i.e., $\|D_i - D_l\| \leq \alpha \|\boldsymbol{x}_i - \boldsymbol{x}_l\|_2$. We can prove the following theorem.

***Theorem** 2:* Let p be the output function of LWkNN-LDL, and f be the derived classifier as Eq. (2). For any $\delta > 0$, the following bound holds with probability at least $1 - \delta$:

$$\mathbb{E}_{\boldsymbol{x}} \left[\sum_{j=1}^{m} |d_{\boldsymbol{x}}^{y_j} - p_{\boldsymbol{x}}^{y_j}| \right] \leq \frac{4\alpha k \sqrt{q}}{\delta} \left(\frac{2k}{n} \right)^{1/q+1}, \tag{14}$$

and the next bound holds with probability at least $1 - \delta$:

$$L(f) - L^* \leq \frac{4\alpha k \sqrt{q}}{\delta} \left(\frac{2k}{n} \right)^{1/q+1}. \tag{15}$$

Proof: Let $\boldsymbol{x}'_l$ be the lth nearest neighbor of $\boldsymbol{x}$ and D'_l be its label distribution. Then, $p(\boldsymbol{x}) = \sum_l w_l \cdot D'_l$, which yields

$$\|D - p(\boldsymbol{x})\|_1 = \left\| D - \sum_l w_l \cdot D'_l \right\|_1 \leq \sum_l w_l \cdot \|D - D'_l\|_1.$$

By the α-Lipschitzness of the given label distribution, we have

$$\|D - p(\boldsymbol{x})\|_1 \leq \alpha \sum_l w_l \cdot \|\boldsymbol{x} - \boldsymbol{x}'_l\| \leq \alpha \sum_l \|\boldsymbol{x} - \boldsymbol{x}'_l\|_2$$

The second inequality holds because we clip and normalize w_l to satisfy $w_l \geq 0$ and $\sum_l w_l = 1$. Taking expectation on both sides of the above equation, we have

$$\mathbb{E}_{\boldsymbol{x},S}\left[\sum_j |d_{\boldsymbol{x}}^{y_j} - p_{\boldsymbol{x}}^{y_j}|\right] \leq \alpha \cdot \mathbb{E}_{\boldsymbol{x},S}\left[\sum_l \|\boldsymbol{x} - \boldsymbol{x}'_l\|_2\right] \leq 4\alpha\sqrt{q}k\left(\frac{2k}{n}\right)^{1/q+1},$$

where the second inequality is by [37]. We finish the proof of Eq. (14) by applying the Markov's inequality to the above inequality. Additionally, combining Eqs. (4) and (14), we complete the proof for Eq. (15). ■

Theorem 2 also holds for LDkNN-LDL and can be proven in a similar way. In Eq. (14), the left-hand side is the expected L_1-norm loss, and the right-hand side approaches 0 if n goes to infinity. In addition, Eq. (15) shows that the classification error approaches the Bayes error if n increases to infinity. Theorem 2 indicates that LWkNN-LDL and LDkNN-LDL can learn any general-form label distribution, and their classification error may approach the Bayes error, under a mild assumption of Lipschitzness, which justifies the theoretical advantages of LWkNN-LDL and LDkNN-LDL.

VI. Experiments

A. Methodology

1) Datasets: We conduct extensive experiments on 17 real-world datasets with label distribution, the statistics of which are summarized in Table II. The average margin (the last column) will be introduced in Section VI-D3.

The first thirteen datasets are collected by Geng [4]. In detail, the first nine datasets (from Alpha to Spo5) are about the genome-wide expression of Yeast *Saccharomyces cerevisiae* [43]. The SJAFFE and SBU_3DFE are obtained from two facial expression databases JAFFE [3] and BU_3DFE [44] by normalizing the mean ratings into label distribution. The Scene is a database of natural-scene images whose ranking information is transformed into compatible label distribution [45]. The Movie is about user ratings on movies [26].

Both the M2B [46] and SCUT_FBP [47] are about facial beauty perception. We preprocess the features and label distribution of M2B and SCUT_FBP as [14].

The Twitter_ldl and Flickr_ldl are about visual sentiment [13]. Similar to [22], we first borrow a trained VGGNet [48] to extract 512-D features and then use the principal component analysis to reduce them to 200-D.

In the sequel, we denote each dataset by its first three letters (Spo5 is denoted by Spo5 to distinguish itself from Spo).

2) Evaluation Metrics: We evaluate the classification performance of the comparing methods by two evaluation metrics. First, we regard the optimal label in label distribution as the ground-truth label and evaluate the classification performance of the comparing methods by the 0/1 loss. For example, given a test instance $\boldsymbol{x}_i$, the 0/1 loss is defined by $\mathbb{I}(f(\boldsymbol{x}_i) \neq y_{\ell_i})$. Additionally, we evaluate the generalization of the comparing methods by the error probability as adopted in [18]. Suppose the given label distribution is the condition probability distribution, then the error probability is defined as the expectation of the 0/1 loss w.r.t. the random label variable y, i.e.,

$$\mathbb{E}_y\left[\mathbb{I}(f(\boldsymbol{x}_i) \neq y)\right] = \mathbb{P}(f(\boldsymbol{x}_i) \neq y \mid \boldsymbol{x}_i) = 1 - \mathbb{P}(y = f(\boldsymbol{x}_i) \mid \boldsymbol{x}_i) = 1 - d_{\boldsymbol{x}_i}^{f(\boldsymbol{x}_i)}.$$

The prediction of the label with a higher label description degree has smaller error probability. For example, given a test instance with label distribution $[0.3, 0.5, 0.2, 0.0]$, y_2, y_1, and y_3 have the highest, second- and third-highest label description degrees, respectively. The error probability of predicting $y_1, y2, y3$, and y_4 are $0.7, 0.5, 0.8$, and 1.0, respectively.

3) Comparing Methods: We compare LWkNN-LDL and LDkNN-LDL against two representative LDL baselines, AA-kNN and SA-BFGS[2] [4], two state-of-the-art LDL approaches, LDL-SCL[3] [21] and LDL-LDM[4] [22], and two specially designed LDL algorithms for classification, RWLM-LDL [17] and LDLM [22]. The details of these methods are as below.

1) AA-kNN [4]: For a test instance, it calculates the mean of the label distribution of its k-nearest neighbors as the predicted label distribution.
2) SA-BFGS [4]: It applies the maximum entropy model to learn label distribution and uses the KL divergence as the learning metric.
3) LDL-SCL [21]: It exploits the local label correlations of LDL, which are encoded into extra features. Then, it jointly learns label distribution and the encoding of label correlations.

TABLE II
Characteristics of the experimental datasets.

ID	Dataset	*#Examples*	*#Features*	*#Labels*	avg. margin
1	Alpha	2,465	24	18	0.004
2	Cdc	2,465	24	15	0.005
3	Cold	2,465	24	4	0.034
4	Diau	2,465	24	7	0.014
5	Dtt	2,465	24	4	0.022
6	Elu	2,465	24	14	0.004
7	Heat	2,465	24	6	0.021
8	Spo	2,465	24	6	0.031
9	Spo5	2,465	24	3	0.083
10	SJAFFE	213	243	6	0.257
11	SBU_3DFE	2,500	243	6	0.090
12	Scene	2,000	294	9	0.131
13	Movie	7,755	1,869	5	0.104
14	M2B	1,240	250	5	0.513
15	SCUT_FBP	1,500	300	5	0.291
16	Twitter_ldl	10,040	200	8	0.369
17	Flickr_ldl	11,150	200	8	0.490

[2]Code:http://ldl.herokuapp.com/download
[3]Code:https://github.com/NJUST-IDAM/LDL-SCL
[4]Code:https://github.com/wangjing4research/LDL-LDM

4) LDL-LDM [22]: It considers both global and local label correlations of LDL by learning global and local label distribution manifolds.
5) RWLM-LDL [17]: It re-weights instances w.r.t. the entropy of label distribution and borrows large margin to address the objective inconsistency.
6) LDLM [18]: It establishes the label distribution margin theory for LDL and applies the theory to address the objective inconsistency.

We replace the kNN algorithm of AA-kNN with the distance-weighted kNN method [30] and derive DAA-kNN. Here, AA-kNN, DAA-kNN, SA-BFGS, LDL-SCL, and LDL-LDM are five general-purpose LDL methods, which aim to learn label distribution. RWLM-LDL and LDLM are two specialized LDL methods for classification.

We set the parameters of the comparing methods as follows. For AA-kNN and DAAkNN, k is tuned from 1 to 21. For SA-BFGS, the default parameters are used. We tune the parameters of LDL-SCL, LDL-LDM, and LDLM as suggested by [21], [22], and [18], respectively. For RWLM-LDL, $\lambda_1 = 0.0001$, $\lambda_2 = 1$, and $\rho = 0.1$, as suggested in [17]. For LWkNN-LDL and LDkNN-LDL, $\lambda_1 = 0.001$, $\lambda_2 = 1$, $\rho = 0.1$, and k is tuned from 11 to 21. We borrow the Manopt toolbox [49] to implement the steepest descent [38] to update the parameters of LWkNN-LDL and LDkNN-LDL. We tune the parameters of each method on the training set by five-fold cross validation. In addition, we run the comparing algorithms on a Linux server with 2.70GHz CPU and 62GB memory.

B. Classification Results and Discussion

We run each method with ten times random data partitions (90% for training and 10% for testing). Tables III and IV tabulate the mean classification results (mean±std %) for each comparing method on 17 datasets in terms of 0/1 loss and error probability, respectively. We highlight the best results in boldface and underline the second-best ones. Besides, we conduct the pairwise t-tests and use $\bullet/\circ$ to indicate whether LDkNN-LDL is significantly better/worse than the comparing methods (at a confidence level of 0.05). We also summarize the top-1 times, average ranks, and win/tie/loss (W/T/L) counts in the last three rows of Tables III and IV.

From Tables III and IV, LDkNN-LDL ranks first in 70.59% (12 out of 17) and 82.35% (14 out of 17) in terms of 0/1 loss and error probability, respectively. It statistically outperforms other comparing methods in 77.20% (105 wins out of 136 pairwise t-tests) for both 0/1 loss and error probability. LDkNN-LDL and LWkNN-LDL achieve the best and second-best mean performance (i.e., average rank), respectively. Besides, we can make the following four observations from Tables III and IV:

1) LWkNN-LDL achieves the second-best performance in most of the datasets. LDkNN-LDL achieves statistically better or comparable performance against LWkNN-LDL because it further considers the difference of neighborhoods by learning distance-dependent weights.
2) RWLM-LDL, LDLM, LWkNN-LDL, and LDkNN-LDL beat the other five methods by a margin. This is because they are specialized LDL methods for classification, but the other five are general-purpose LDL methods, which suffer from the objective inconsistency and have worse classification performance.
3) LDL-LDM achieves poor classification performance. Although it exploits global and local label correlations, it is a general-purpose LDL method facing the objective inconsistency. As a result, our methods statistically outperform LDL-LDM because they are specially designed for classification.
4) LDkNN-LDL statistically outperforms RWLM-LDL and LDLM. Moreover, LWkNN-LDL achieves better mean performance than RWLM-LDL and LDLM. The reason may lie in that LDkNN-LDL and LWkNN-LDL apply the kNN algorithm to LDL which can learn any form of label distribution.
5) LDkNN-LDL and LWkNN-LDL remarkably improve DAA-kNN and AA-kNN in terms of both 0/1 loss and error probability. The reason is that they learn weights in kNN algorithm to learn label distribution and implement large margin, which adapts to the given label distribution and improves the classification performance.

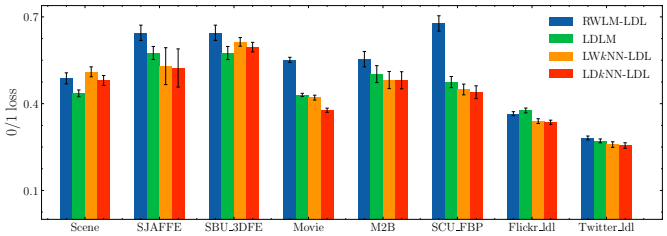


Fig. 5. Comparison results of RWLM-LDL, LDLM, LWkNN-LDL, and LDkNN-LDL (70% samples for training and 30% for testing).

To further study the classification performance of LWkNN-LDL and LDkNN-LDL, we compare them with RWLM-LDL and LDLM with reduced training samples. We run these four approaches for ten times random data partitions on the last eight datasets with 70% samples for training and 30% samples for testing. Fig. 5 reports the mean performance in terms of 0/1 loss. From Fig. 5, LWkNN-LDL and LDkNN-LDL outperform RWLM-LDL and LDLM by a margin.

To summarize, the experimental results justify the advantages of LWkNN-LDL and LDkNN-LDL for classification.

C. Analysis of Learning Weighted kNN and Large Margin

Our proposed methods have two important components, i.e., learning weighted kNN and large margin. In this subsection, we conduct more analysis to investigate their advantages.

1) Usefulness of Learning Weighted kNN for LDL: To start, we call models (7) and (10) WkNN-LDL and DkNN-LDL, respectively. They learn weighted kNN for LDL and are two general-purpose LDL methods. To validate their advantages, we compare them with the five general-purpose LDL methods of Section VI-A3. Following the same evaluation protocol as [22], we run the comparing methods with ten times random data partitions (60% for training and 40% for testing) and evaluate their performance in terms of two widely used LDL metrics, KL divergence (the smaller the better) and cosine

TABLE III
THE EXPERIMENTAL RESULTS (MEAN±STD %) IN TERMS OF 0/1 LOSS. WE CONDUCT THE PAIRWISE t-TESTS AND USE •/○ TO INDICATE WHETHER LDkNN-LDL IS STATISTICALLY BETTER/WORSE THAN EACH COMPARING METHOD (AT 0.05 SIGNIFICANCE LEVEL). BESIDES, WE HIGHLIGHT THE BEST RESULTS IN BOLDFACE, UNDERLINE THE SECOND-BEST RESULTS, AND SUMMARIZE THE TOP-1 TIMES, AVG. RANKS, AND WIN/TIE/LOSS COUNTS OF EACH METHOD IN THE LAST THREE ROWS.

Dataset	DAA-kNN	AA-kNN	SA-BFGS	LDL-SCL	LDL-LDM	RWLM-LDL	LDLM	LWkNN-LDL	LDkNN-LDL
Alp	88.60±2.28•	88.27±2.01•	89.82±2.34•	91.24±1.73•	89.49±2.50•	78.70±2.34○	**78.34±3.66**○	84.83±1.67•	83.93±1.34
Cdc	83.57±1.56•	83.12±1.40•	82.31±2.11•	82.68±2.26•	82.31±2.13•	81.78±2.20	81.62±2.87	80.81±1.78•	**80.00±1.85**
Col	56.18±4.07•	56.43±4.25•	57.89±3.29•	57.81±3.37•	57.85±3.64•	57.53±3.00•	56.71±2.61•	53.22±3.86	**53.10±3.23**
Dia	69.01±2.49•	68.64±3.75•	69.62±3.88•	70.23±3.72•	69.29±3.62•	65.85±3.63•	65.27±1.09	65.93±2.56	**64.79±2.43**
Dtt	63.28±3.77•	63.16±2.87•	62.84±2.08•	63.49±2.29•	62.92±1.87•	62.68±2.72•	62.43±2.75•	59.19±2.86•	**58.22±2.82**
Elu	86.45±2.63•	86.69±2.25•	90.31±2.02•	90.91±2.17•	90.22±1.77•	**79.90±3.35**○	80.32±2.55	83.45±1.72	83.37±1.50
Hea	67.59±2.56•	69.17±2.67•	69.90±2.71•	69.62±2.52•	69.58±2.52•	66.94±2.21•	66.66±2.76•	64.91±1.68	**63.82±2.36**
Spo	58.99±3.44•	55.91±3.70•	55.26±3.43	55.87±3.52	55.62±3.53	54.73±1.89	54.69±3.29	**54.49±3.51**	55.09±3.03
Spo5	52.41±3.77•	54.48±3.31•	57.04±3.00•	59.23±4.02•	55.21±2.76•	52.43±3.05•	52.82±2.04•	49.78±3.45	**48.97±2.53**
SJA	48.35±9.37•	48.85±8.26•	52.14±9.10•	75.15±7.82•	48.29±9.93•	40.40±9.90•	38.96±10.5•	34.26±5.80	**32.42±4.59**
SBU	60.64±3.47•	61.20±3.73•	53.64±2.13○	**52.20±2.95**○	57.32±2.99	56.92±2.77	54.92±3.29	58.00±3.48	57.96±3.46
Sce	52.85±2.21•	52.85±2.21•	61.00±2.70•	66.60±4.24•	57.55±3.66•	43.45±2.37○	**41.35±3.53**○	52.00±1.96•	51.25±1.79
Mov	42.54±2.29•	42.49±1.72•	42.05±1.45•	42.85±1.12•	44.59±1.77•	40.86±1.56	41.10±1.94•	40.66±2.14	**40.35±1.91**
M2B	47.50±3.56•	48.23±3.36•	50.73±4.50•	48.15±2.47•	51.61±5.83•	48.06±3.04•	46.61±2.60•	44.19±3.12	**43.71±3.42**
SCU	45.47±3.62•	45.40±4.36•	69.33±4.67•	54.13±6.70•	48.13±3.41•	46.53±2.53•	45.80±2.97•	41.53±3.65	**41.07±2.92**
Fli	33.62±1.19•	34.24±1.37•	36.05±1.32•	44.73±1.30•	35.24±1.12•	34.65±1.22•	34.97±0.87•	32.98±1.30•	**32.42±1.18**
Twi	25.05±0.97•	25.37±0.93•	26.03±1.32•	27.53±1.08•	25.69±1.19•	25.55±1.03•	26.22±1.09•	24.65±1.06•	**24.17±0.79**
Top-1	0	0	0	1	0	1	2	1	12
Avg. rank	5.53	5.88	7.12	7.82	6.76	3.94	3.41	2.70	1.77
W/T/L	17/0/0	17/0/0	15/1/1	15/1/1	15/2/0	10/4/3	10/5/2	6/11/0	

TABLE IV
THE EXPERIMENTAL RESULTS (MEAN±STD %) IN TERMS OF ERROR PROBABILITY.

Dataset	DAA-kNN	AA-kNN	SA-BFGS	LDL-SCL	LDL-LDM	RWLM-LDL	LDLM	LWkNN-LDL	LDkNN-LDL
Alp	94.29±0.05•	94.28±0.04•	94.28±0.03•	94.30±0.03•	94.27±0.04•	94.26±0.02•	94.25±0.04•	94.22±0.04	**94.22±0.03**
Cdc	92.93±0.05•	92.90±0.05•	92.87±0.05	92.88±0.06	92.88±0.06	92.87±0.05	92.87±0.05	92.86±0.05	**92.85±0.05**
Col	72.99±0.31•	72.99±0.34•	72.98±0.29•	72.98±0.30•	73.00±0.31•	72.96±0.31	72.96±0.33•	**72.77±0.28**	**72.77±0.28**
Dia	84.24±0.12•	84.22±0.17•	84.29±0.16•	84.29±0.16•	84.28±0.16•	84.28±0.10•	84.27±0.13•	84.10±0.15	**84.09±0.11**
Dtt	74.13±0.19•	74.13±0.17•	74.18±0.17•	74.19±0.15•	74.16±0.16•	74.12±0.21•	74.09±0.21•	**73.89±0.13**	73.89±0.12
Elu	92.60±0.06•	92.60±0.06•	92.61±0.06•	92.61±0.05•	92.61±0.05•	92.60±0.05•	92.60±0.04•	92.54±0.04	**92.53±0.04**
Hea	82.30±0.18•	82.37±0.17•	82.42±0.20•	82.42±0.18•	82.42±0.20•	82.33±0.18•	82.29±0.19•	82.12±0.19	**82.07±0.19**
Spo	81.20±0.42•	81.04±0.44•	81.04±0.42•	81.05±0.43•	81.07±0.40•	81.00±0.41	81.00±0.43•	**80.86±0.43**	80.90±0.36
Spo5	64.34±0.60•	64.70±0.55•	65.42±0.50•	65.39±0.43•	65.39±0.47•	65.26±0.58•	64.97±0.67•	63.88±0.45•	**63.74±0.44**
SJA	76.40±1.67•	76.89±1.66•	77.56±1.91•	81.96±1.79•	76.11±2.23•	75.73±2.04	75.74±1.28	75.07±1.22•	**74.47±1.52**
SBU	78.48±0.72•	78.57±0.71•	76.30±0.42○	**75.97±0.43**○	77.25±0.70○	77.34±0.54	76.92±0.63	77.84±0.71	77.79±0.75
Sce	66.29±2.35•	66.45±2.74•	65.81±2.40•	67.82±3.42•	65.52±2.00	64.50±1.58	65.05±2.42	64.87±2.56	**64.38±2.48**
Mov	67.74±0.41•	67.78±0.31•	67.68±0.27•	67.88±0.26•	68.11±0.29•	67.43±0.30	67.58±0.33•	67.43±0.36	**67.36±0.32**
M2B	53.68±2.69•	54.12±2.56•	55.32±3.09•	54.08±2.26•	56.22±3.58•	53.58±2.40	53.54±1.86•	51.78±2.54•	**51.32±2.84**
SCU	53.71±1.17•	53.74±1.55•	71.17±3.42•	58.81±3.07•	55.44±1.43•	54.14±1.41•	54.02±1.15•	52.69±1.56	**52.59±1.38**
Fli	52.61±0.58•	52.95±0.55•	53.50±0.72•	56.97±0.63•	53.24±0.64•	53.19±0.54•	53.20±0.51•	52.35±0.63•	**52.10±0.64**
Twi	41.11±0.72•	41.35±0.81•	41.63±0.96•	42.50±0.75•	41.51±0.93•	41.55±0.86•	41.78±0.71•	41.01±0.81•	**40.74±0.67**
Top-1	0	0	0	1	0	0	0	3	14
Avg. rank	5.59	5.94	6.94	7.47	6.82	4.47	4.06	2.24	1.47
W/T/L	17/0/0	17/0/0	15/1/1	15/1/1	14/2/1	9/8/0	13/4/0	5/12/0	

TABLE V
EXPERIMENTAL RESULTS (MEAN) IN TERMS OF KL DIVERGENCE.

	AA	DAA	SA	SCL	LDM	WkNN	DkNN
SJA	0.0658	0.0655	0.0680	0.0526	**0.0404**	0.0566	0.0562
SBU	0.0760	0.0732	0.0598	0.0626	**0.0537**	0.0722	0.0723
Sce	0.8786	2.8248	0.9568	0.8331	**0.7624**	0.8450	0.7965
Mov	0.1078	0.1118	0.1673	1.0527	**0.0984**	0.1091	0.1052
M2B	0.5715	0.5159	0.9542	0.5066	**0.4952**	0.5422	0.5243
SCU	0.4973	3.4461	2.1339	0.6894	0.6877	0.3842	**0.3736**
Fli	0.7458	7.2173	0.4963	0.6910	0.4907	0.4844	**0.4798**
Twi	1.1837	7.5112	0.5387	0.6787	0.5281	0.5458	**0.5262**
avg. rank	5.38	5.88	5.13	3.75	1.88	3.25	2.76

TABLE VI
EXPERIMENTAL RESULTS (MEAN) IN TERMS OF COSINE SIMILARITY.

	AA	DAA	SA	SCL	LDM	WkNN	DkNN
SJA	0.9371	0.9371	0.9388	0.9501	**0.9620**	0.9462	0.9464
SBU	0.9254	0.9261	0.9422	0.9380	**0.9472**	0.9291	0.9290
Sce	0.7185	0.6140	0.6836	0.7209	**0.7438**	0.7038	0.7239
Mov	0.9291	0.9267	0.9003	0.9178	**0.9350**	0.9284	0.9287
M2B	0.7477	0.7636	0.6750	0.7657	**0.7729**	0.7546	0.7582
SCU	0.8383	0.7579	0.6270	0.6652	0.6663	0.8415	**0.8431**
Fli	0.8376	0.7470	0.8334	0.7700	0.8357	0.8427	**0.8462**
Twi	0.8536	0.7653	0.8505	0.8226	0.8543	0.8550	**0.8589**
avg. rank	4.50	5.75	5.50	4.00	2.25	3.38	2.63

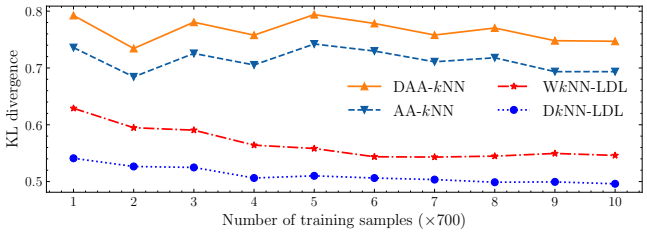


Fig. 6. Performance of DAAkNN, AA-kNN, WkNN-LDL, and DkNN-LDL with an increasing number of training samples from Flickr_ldl.

TABLE VII
EXPERIMENTAL RESULTS (WIN/TIE/LOSS COUNTS) OF THE PAIRWISE t-TESTS AT A CONFIDENCE LEVEL OF 0.05.

Metric	LWkNN-LDL *against*	LDkNN-LDL *against*
	WkNN-LDL	DkNN-LDL
0/1 loss	**17/0/0**	**17/0/0**
Error prob.	**16/1/0**	**17/0/0**

similarity (the larger the better) [4]. Tables V and VI report the mean results on the last eight datasets in Table II[5]. From Tables V and VI, LDL-LDM achieves better performance than DkNN-LDL and WkNN-LDL because it exploits both global and local label correlations, which are especially helpful for LDL. Nevertheless, it is a general-purpose LDL method that has poor classification performance, as reported in Section VI-B. Besides, DkNN-LDL and WkNN-LDL outperform AA-kNN and DAA-kNN by a margin, which justifies the usefulness of learning weighted kNN for LDL.

Next, we study the performance of DAA-kNN, AA-kNN, WkNN-LDL, and DkNN-LDL with different number of training samples. First, we randomly split Flickr_ldl into a training set (7000 samples) and a test set (4150 samples). Then, we run these four methods ($k = 11$) with an increasing number of training samples from the training set and evaluate their performance on the test set. Fig. 6 presents the results. DkNN-LDL and WkNN-LDL achieve remarkably better performance than DAA-kNN and AA-kNN. In particular, WkNN-LDL and DkNN-LDL with only 700 training samples even outperform DAA-kNN and AA-kNN with 7000 training samples. The reason lies in that WkNN-LDL and DkNN-LDL can better adapt to the given label distribution than AA-kNN and DAA-kNN through learning data-dependent weights, especially for small training data.

2) Usefulness of Large Margin for Classification: LWkNN-LDL and LDkNN-LDL differ from WkNN-LDL and DkNN-LDL in using large margin. To justify the advantage of that, we run these four approaches with $k = 11$ on all datasets and then compare their performance. Figs. 7 and 8 show some typical examples. Fig. 9 reports the detailed results on the last eight datasets. Moreover, we conduct the pairwise t-tests and summarize the counts of win/tie/loss (at a confidence level of 0.05) in Table VII.

For all examples shown in Figs. 7 and 8, WkNN-LDL and DkNN-LDL achieve smaller L_1-norm loss than LWkNN-LDL and LDkNN-LDL, respectively, but neglect the optimal labels. LWkNN-LDL and LDkNN-LDL benefit from large margin whose predictions equal the optimal labels. Fig. 9 shows that LWkNN-LDL and LDkNN-LDL outperform WkNN-LDL and DkNN-LDL. According to Table VII, LWkNN-LDL and LDkNN-LDL have statistically better performance than WkNN-LDL and DkNN-LDL (67 wins out of 68 pairwise t-tests). Large margin helps solve the objective inconsistency and improve the classification performance of LWkNN-LDL and LDkNN-LDL.

D. Further Analysis

1) Convergence: Fig. 10 plots the changes of the objective functions of LWkNN-LDL and LDkNN-LDL w.r.t. the number of iterations on Flickr_ldl and Twitter_ldl. As shown in Fig. 10, the objective functions of LWkNN-LDL and LDkNN-LDL converge after about 20 iterations and reach stable values, which validates the effect of the steepest descent for solving the optimization problems.

2) Running Time: Table VIII reports the mean running time of LDkNN-LDL, LWkNN-LDL, RWLM-LDL, and LDLM on the three largest datasets for ten times random data partitions (50% for training and 50% for testing). From Table VIII, LWkNN-LDL and LDkNN-LDL spend less training time and more testing time than RWLM-LDL and LDLM because they

[5]We denote SA-BFGS by SA, and leave out the prefix/suffix of "LDL" and "kNN" in the name of each method.

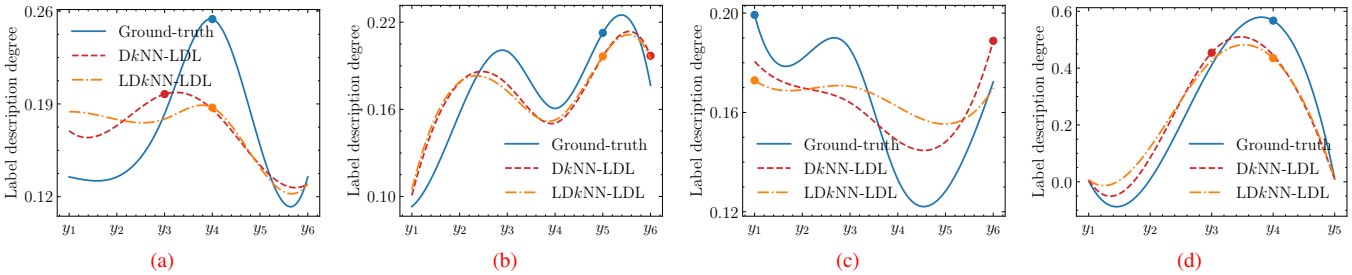


Fig. 7. Comparison between LWkNN-LDL and WkNN-LDL on some typical examples of (a) SBU_3DFE, (b) SJAFFE, (c) Spo, and (d) M2B. We highlight the highest (predicted) label description degrees. The Y-axis denotes label description degree. For all examples, WkNN-LDL has smaller L_1-norm losses than LWkNN-LDL but neglects the optimal label. However, the predictions of LWkNN-LDL equal the optimal labels for all examples.

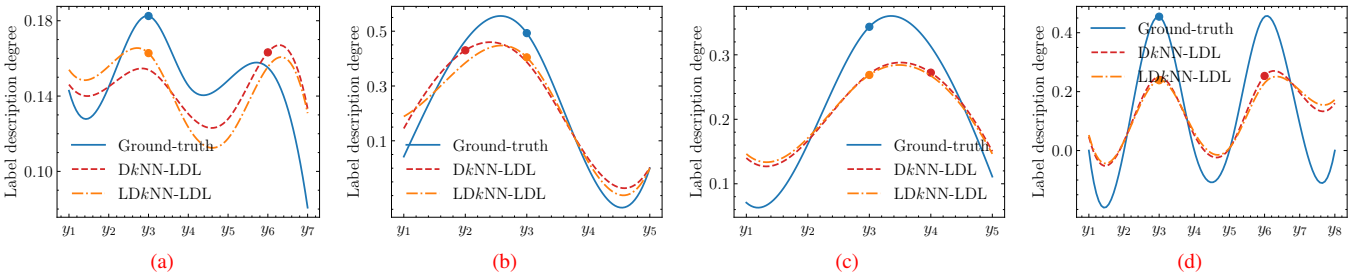


Fig. 8. Comparison between LDkNN-LDL and DkNN-LDL on some typical examples of (a) Diau, (b) SCUT_FBP, (c) Movie, and (d) Flickr_ldl. The Y-axis represents label description degree. For all examples, DkNN-LDL achieves smaller L_1-norm losses than LDkNN-LDL but misses the optimal label. However, LDkNN-LDL successfully outputs the optimal labels for all examples.

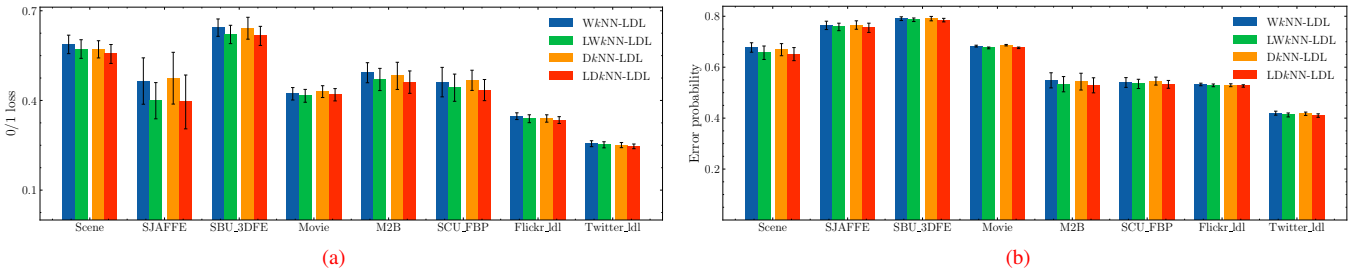


Fig. 9. Detailed results of WkNN-LDL, LWkNN-LDL, DkNN-LDL, and LDkNN-LDL in terms of (a) 0/1 loss and (b) error probability.

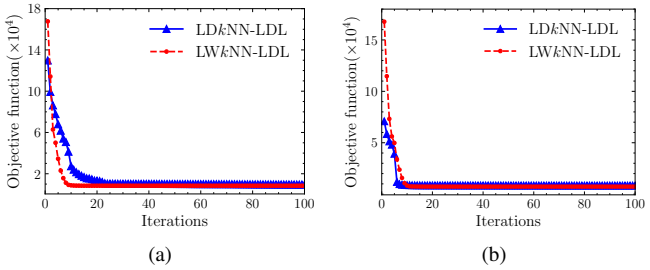


Fig. 10. Convergence on (a) Flickr_ldl and (b) Twitter_ldl.

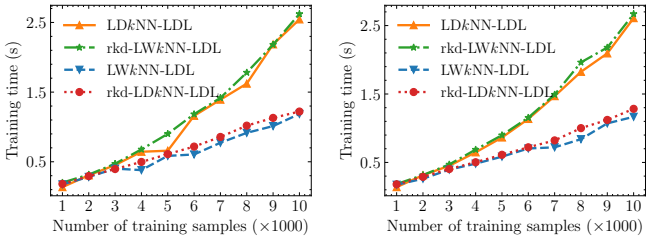
TABLE VIII
RUNNING TIME (S) ON THE THREE LARGEST DATASETS.

Algorithm	Movie			Twitter_ldl			Flickr_ldl		
	train	test	total	train	test	total	train	test	total
RWLM-LDL	11.07	0.02	11.09	6.81	0.00	6.81	8.04	0.00	8.04
LDLM	6.75	0.01	6.76	3.41	0.00	3.42	3.74	0.00	3.74
LWkNN-LDL	0.72	0.51	1.22	0.89	0.38	1.27	1.02	0.47	1.49
LDkNN-LDL	0.74	0.50	1.24	0.91	0.38	1.29	1.08	0.48	1.56

need to search the k-nearest neighbors for test instances in the test phase. Moreover, LWkNN-LDL and LDkNN-LDL need less total running time than RWLM-LDL and LDLM.

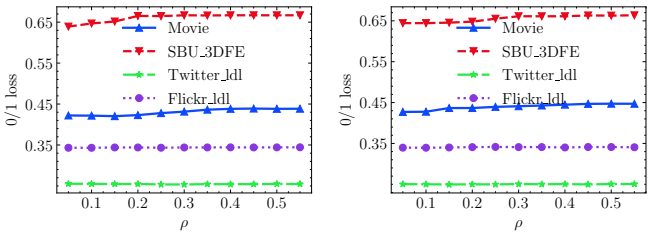
Next, we borrow the randomized kd-tree algorithm [42] to search for k-nearest neighbors for LWkNN-LDL and LDkNN-LDL (denoted by the prefix "rkd-"). Fig. 11 shows the training time of these methods with an increasing number of training samples. Fig. 11 shows that LWkNN-LDL and LDkNN-LDL need quadratic running time, which can be reduced to almost linear by the randomized kd-tree approach. Thus, we can use approximate searching rather than brute-force searching in our methods for large-scale datasets.

3) Analysis of Margin: To start, we analyze the margin of the given label distribution. Define the margin of the label



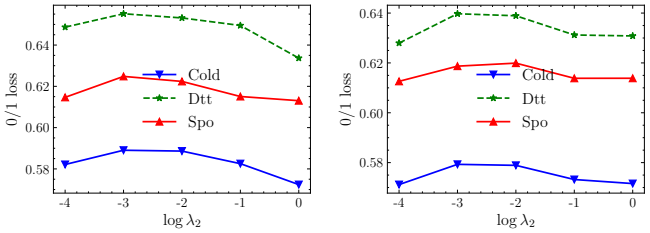
(a) (b)

Fig. 11. Training time on (a) Flickr_ldl and (b) Twitter_ldl with an increasing number of training samples.



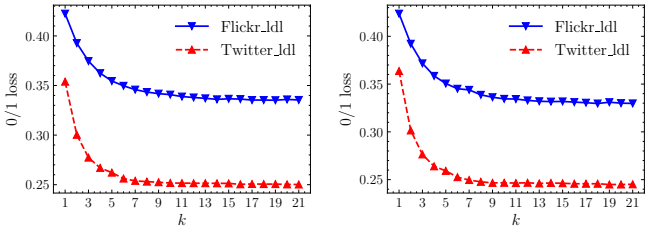
(a) (b)

Fig. 12. Performance of (a) LWkNN-LDL and (b) LDkNN-LDL with ρ varying from 0.05 to 0.55 with a step of 0.05.



(a) (b)

Fig. 13. Performance of (a) LWkNN-LDL and (b) LDkNN-LDL with λ_2 varying from {1e-4, 1e-3, 1e-2, 1e-1, 1e0} on Cold, Dtt, and Spo.



(a) (b)

Fig. 14. Performance of (a) LWkNN-LDL and (b) LDkNN-LDL with k ranging from 1 to 21 on Twitter_ldl, and Flickr_ldl.

distribution for $\boldsymbol{x}_i$ by

$$\rho_i = d_{\boldsymbol{x}_i}^{y_{\ell_i}} - \max_{j \neq \ell_i} d_{\boldsymbol{x}_i}^{y_j},$$

i.e., the difference between the highest and the second-highest label description degrees. Table II summarizes the mean margins of all datasets. From Table II, the last eight datasets have larger margins than the first nine ones. From Tables II, III, and IV, LWkNN-LDL and LDkNN-LDL work more efficiently on the last eight datasets than on the first nine ones. That is, our methods tend to have better performance on datasets with large label distribution margins.

Next, we study the influence of ρ. We run LWkNN-LDL and LDkNN-LDL with ρ varying from 0.05 to 0.55 and report their performance in Fig. 12. Our methods are sensitive to ρ on datasets with small label distribution margins, like SBU_3DFE and Movie, where small ρ's are preferred because large ones would far exceed the true margins. Besides, our methods are robust to ρ on datasets with large label distribution margins, like Twitter_ldl and Flickr_ldl, where large ρ's are tolerant. Generally, $\rho = 0.1$ leads to satisfying performance for most of the datasets. So, we set $\rho = 0.1$ in the experiments.

4) Parameter Sensitivity: Here we analyze the sensitivity of the trade-off parameter λ_2 and the number of nearest neighbors k. First, we run LWkNN-LDL and LDkNN-LDL with λ_2 from the candidate set $\{$1e-4, 1e-3, $\cdots$, 1e0$\}$. Fig. 13 reports their performance on Cold, Dtt, and Spo. According to Fig. 13, $\lambda_2 = 1$ brings better performance for both LWkNN-LDL and LDkNN-LDL. Second, we run LWkNN-LDL and LDkNN-LDL with k from 1 to 21 to study the sensitivity of k. Fig. 14 shows their performance on Flickr_ldl and Twitter_ldl. According to Fig. 14, LWkNN-LDL and LDkNN-LDL achieve better performance when k is within [11, 21].

VII. Conclusion

In this paper, we propose two novel LDL methods, which are specially designed for classification. Our methods learn weight vectors for the kNN algorithm to learn label distribution and implement large margin to address the objective inconsistency between LDL and classification. The theoretical results disclose that our methods can learn any general-form label distribution and their classification error may approach the Bayes error. In the experiments, our methods significantly outperform the state-of-the-art LDL algorithms.

Our methods apply large margin and encourage a margin ρ in the predicted label distribution. However, ρ is defined independently of the given label distribution, which is inefficiently since it may significantly differs from the true margin of the label distribution. A feasible solution is to set ρ to the mean margin of the label distribution and use different ρ's for different datasets. Also, we can set ρ to the margin of the label distribution (i.e., $\rho = \rho_i$ for $\boldsymbol{x}_i$) and adopt different ρ's for different training samples. In the future, we will further explore how to set ρ. Besides, our methods use the same k for all instances. In the future, we will explore how to learn different k's for different instances [50].

References

[1] J. Peterson, R. Battleday, T. Griffiths, and O. Russakovsky, "Human uncertainty makes classification more robust," in *Proc. 2019 IEEE/CVF Int. Conf. Comput. Vis. (ICCV'19)*, Oct. 2019, pp. 9616–9625.

[2] B. Gao, C. Xing, C. Xie, J. Wu, and X. Geng, "Deep label distribution learning with label ambiguity," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2825–2838, Mar. 2017.

[3] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with Gabor wavelets," in *Proc. 3th IEEE Int. Conf. Auto. Face Gesture Recogn.*, Apr. 1998, pp. 200–205.

[4] X. Geng, "Label distribution learning," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1734–1748, July 2016.

[5] X. Geng, C. Yin, and Z. Zhou, "Facial age estimation by learning from label distributions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 10, pp. 2401–2412, Oct. 2013.

[6] X. Wen, B. Li, H. Guo, Z. Liu, G. Hu, M. Tang, and J. Wang, "Adaptive variance based label distribution learning for facial age estimation," in *Proc. Eur. Conf. Comput. Vis. (ECCV'20)*, Nov. 2020, pp. 379–395.

[7] W. Shen, Y. Guo, Y. Wang, K. Zhao, B. Wang, and A. Yuille, "Deep differentiable random forests for age estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 404–419, Feb. 2021.

[8] S. Chen, J. Wang, Y. Chen, Z. Shi, X. Geng, and Y. Rui, "Label distribution learning on auxiliary label space graphs for facial expression recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. (CVPR'20)*, June 2020, pp. 13 984–13 993.

[9] X. Zhou, Z. Wei, M. Xu, S. Qu, and G. Guo, "Facial depression recognition by deep joint label distribution and metric learning," *IEEE Trans. Affect. Comput.*, Sep 2020, early access, doi:10.1109/TAFFC.2020.3022732.

[10] Y. Shu, P. Yang, N. Liu, S. Zhang, G. Zhao, and Y. Liu, "Emotion distribution learning based on peripheral physiological signals," *IEEE Trans. Affect. Comput.*, Mar. 2022, early access, doi:10.1109/TAFFC.2022.3163609.

[11] X. Geng, X. Qian, Z. Huo, and Y. Zhang, "Head pose estimation based on multivariate label distribution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 1974–1991, 2022.

[12] J. Yang, D. She, and M. Sun, "Joint image emotion classification and distribution learning via deep convolutional neural network," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI'17)*, Aug. 2017, pp. 3266–3272.

[13] J. Yang, M. Sun, and X. Sun, "Learning visual sentiment distributions via augmented conditional probability neural network," in *Proc. 31st AAAI Conf. Artif. Intell. (AAAI'17)*, Feb. 2017, pp. 224–230.

[14] Y. Ren and X. Geng, "Sense beauty by label distribution learning," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI'17)*, Aug. 2017, pp. 2648–2654.

[15] Y. Fan, S. Liu, B. Li, Z. Guo, A. Samal, J. Wan, and S. Z. Li, "Label distribution-based facial attractiveness computation by deep residual learning," *IEEE Trans. Multimedia*, vol. 20, no. 8, pp. 2196–2208, Aug. 2018.

[16] N. Xu, J.-Y. Li, Y.-P. Liu, and X. Geng, "Trusted-data-guided label enhancement on noisy labels," *IEEE Trans. Neural Netw. Learn. Syst.*, Apr. 2022, early access, doi:10.1109/TNNLS.2022.3162316.

[17] J. Wang, X. Geng, and H. Xue, "Re-weighting large margin label distribution learning for classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5445–5459, Sep. 2022.

[18] J. Wang and X. Geng, "Label distribution learning machine," in *Proc. 38th Int. Conf. Mach. Learn. (ICML'21)*, July 2021, pp. 10 749–10 759.

[19] ——, "Learn the highest label and rest label description degrees," in *Proc. 30th Int. Joint Conf. Artif. Intell. (IJCAI'21)*, Aug. 2021, pp. 3097–3103.

[20] T. Ren, X. Jia, W. Li, L. Chen, and Z. Li, "Label distribution learning with label-specific features," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI'19)*, July 2019, pp. 3318–3324.

[21] X. Jia, Z. Li, X. Zheng, W. Li, and S. Huang, "Label distribution learning with label correlations on local samples," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1619–1631, Apr. 2021.

[22] J. Wang and X. Geng, "Label distribution learning by exploiting label distribution manifold," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 2, pp. 839–852, Feb. 2023.

[23] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra, "A maximum entropy approach to natural language processing," *Comput. Linguist.*, vol. 22, no. 1, pp. 39–71, Mar. 1996.

[24] W. Shen, K. Zhao, Y. Guo, and A. L. Yuille, "Label distribution learning forests," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NeurIPS'17)*, Dec. 2017, pp. 834–843.

[25] L. Devroye, L. Györfi, and G. Lugosi, *A probabilistic theory of pattern recognition*. New York, NY, USA: Springer, 1996.

[26] X. Geng and P. Hou, "Pre-release prediction of crowd opinion on movies by label distribution learning," in *Proc. 24th Int. Joint Conf. Artif. Intell. (IJCAI'15)*, July 2015, pp. 3511–3517.

[27] L. Jiang, H. Zhang, F. Tao, and C. Li, "Learning from crowds with multiple noisy label distribution propagation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6558–6568, Nov. 2022.

[28] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan 1967.

[29] G. H. Chen and D. Shah, "Explaining the success of nearest neighbor methods in prediction," *Found. Trends Mach. Learn.*, vol. 10, no. 5-6, pp. 337–588, May 2018.

[30] S. A. Dudani, "The distance-weighted k-nearest-neighbor rule," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, no. 4, pp. 325–327, Apr. 1976.

[31] V. Vapnik and A. Y. Chervonenkis, "A note on one class of perceptrons," *Automat. Rem. Control*, vol. 25, no. 1, pp. 821–837, 1964.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[33] X. Wu and Z. Zhou, "A unified view of multi-label performance measures," in *Proc. 34th Int. Conf. Mach. Learn. (ICML'17)*, July 2017, pp. 3780–3788.

[34] B. Jia and M. Zhang, "Maximum margin multi-dimensional classification," *IEEE Trans. Neural Netw. Learn. Syst.*, June 2021, early access, doi:10.1109/TNNLS.2021.3084373.

[35] J. Chai, I. W. Tsang, and W. Chen, "Large margin partial label machine," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2594–2608, 2020.

[36] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, no. 9, pp. 207–244, Feb. 2009.

[37] J. Wang and X. Geng, "Theoretical analysis of label distribution learning," in *Proc. 33rd AAAI Conf. Artif. Intell. (AAAI'19)*, Jan. 2019, pp. 5256–5263.

[38] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[39] X. Chang, F. Nie, S. Wang, Y. Yang, X. Zhou, and C. Zhang, "Compound rank-k projections for bilinear analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 7, pp. 1502–1513, 2016.

[40] C. Yan, X. Chang, M. Luo, Q. Zheng, X. Zhang, Z. Li, and F. Nie, "Self-weighted robust lda for multiclass classification with edge classes," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 1, pp. 1–19, Dec. 2020.

[41] W. Li, Y. Zhang, Y. Sun, W. Wang, M. Li, W. Zhang, and X. Lin, "Approximate nearest neighbor search on high dimensional data – experiments, analyses, and improvement," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1475–1488, Aug. 2020.

[42] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2227–2240, Nov. 2014.

[43] M. B. Eisen, P. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 95, no. 25, pp. 14 863–14 868, Dec. 1998.

[44] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, "A 3D facial expression database for facial behavior research," in *Proc. 7th Int. Conf. Autom. Face Gesture Recogn.*, Apr. 2006, pp. 211–216.

[45] X. Geng and L. Luo, "Multilabel ranking with inconsistent rankers," in *Proc. 2014 IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR'14)*, June 2014, pp. 3742–3747.

[46] T. V. Nguyen, S. Liu, B. Ni, J. Tan, Y. Rui, and S. Yan, "Sense beauty via face, dressing, and/or voice," in *Porc. 20th ACM Int. Conf. Multimedia (MM'12)*, Oct. 2012, pp. 239–248.

[47] D. Xie, L. Liang, L. Jin, J. Xu, and M. Li, "SCUT-FBP: A benchmark dataset for facial beauty perception," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Oct. 2015, pp. 1821–1826.

[48] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Repr.*, May 2015, pp. 1–14.

[49] N. Boumal, B. Mishra, P. Absil, and R. Sepulchre, "Manopt, a Matlab toolbox for optimization on manifolds," *J. Mach. Learn. Res.*, vol. 15, no. 42, pp. 1455–1459, 2014.

[50] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient knn classification with different numbers of nearest neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1774–1785, May 2018.

Jing Wang received the B.Sc. degree in computer science from Suzhou University of Science and Technology, Suzhou, China, in 2013, and the M.Sc. degree in computer science from Northeastern University, Shenyang, China, in 2015, and the Ph.D. degree in software engineering from Southeast University, Nanjing, China, in 2021. He is currently an assistant professor of the School of Computer Science and Engineering, Southeast University, Nanjing. His research interests include pattern recognition and machine learning.

Xin Geng (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science from Nanjing University, Nanjing, China, in 2001 and and 2004, respectively, and the Ph.D. degree in computer science from Deakin University, Geelong, VIC, Australia, in 2008.

He is currently a chair professor of the School of Computer Science and Engineering, Southeast University, Nanjing. His research interests include machine learning, pattern recognition, and computer vision. He has published over 100 refereed articles in these areas, including those published in prestigious journals and top international conferences.

Dr. Geng has been an Associate Editor of IEEE TRANSACTIONS ON MULTIMEDIA, Frontiers of Computer Science, and Mathematical Foundations of Computing, a Steering Committee Member of Pacific Rim International Conferences on Artificial Intelligence (PRICAI), a Program Committee Chair for conferences, such as PRICAI 2018 and Vision And Learning SEminar (VALSE) 2013, the Area Chair for conferences, such as Computer Vision and Pattern Recognition (CVPR), ACM Multimedia, and Chinese Conference on Pattern Recognition (CCPR), and a Senior Program Committee Member for conferences, such as International Joint Conference on Artificial Intelligence (IJCAI), AAAI Conference on Artificial Intelligence (AAAI), and European Conference on Artificial Intelligence (ECAI). He is a Distinguished Fellow of International Engineering and Technology Institute.