Instance-specific Loss-weighted Decoding for Decomposition-based Multi-class Classification

Bin-Bin Jia, Jun-Ying Liu, and Min-Ling Zhang, *Senior Member, IEEE*

***Abstract*—Multi-class classification problems are often addressed by decomposing them into a set of binary classification tasks. A critical step in this approach is the effective aggregation of predictions from each decomposed binary classifier to yield the final multi-class prediction, a process known as decoding. Existing studies have ignored the varying generalization ability of each binary classifier across different samples during decoding, potentially leading to suboptimal performance. In this paper, we propose an instance-specific loss-weighted decoding strategy that gauges the generalization ability of each binary classifier for one specific sample based on its neighboring samples. This estimated generalization ability is then utilized to adjust the importance of the binary classifier in determining the sample's final prediction. Experimental results validate the effectiveness of the instance-specific loss-weighted decoding strategy. Furthermore, we demonstrate that softmax regression can be reinterpreted as a one-vs-rest decomposition-based multi-class classification algorithm, enabling the application of our decoding strategy to enhance its performance. Comparative studies clearly demonstrate the superiority of the improved softmax regression over its traditional counterpart.**

***Index Terms*—machine learning, multi-class classification, decomposition-based strategy, loss-based decoding**

I. Introduction

MULTI-CLASS classification is one of the most significant and commonly used learning tasks in artificial intelligence and machine learning [69], [78]. A diverse range of real-world applications in various fields, such as bioinformatics [18], [61], text mining [44], [26] and computer vision [62], [64], [36], can be addressed by formulating them within the multi-class classification framework. Moreover, some complicated classification tasks, such as multi-label classification [73], [42] and multi-dimensional classification [23], [24], [25], can also be solved by reducing them into one or a set of multi-class classification problem(s). Compared with binary classification, which aims to classify objects into one of only two potential classes, multi-class classification involves a greater number of possible classes, rendering it more challenging to solve [7], [57], [76], [67], [31], [46].

The existing multi-class classification methods can be roughly divided into two categories: direct strategy and indirect strategy [38]. The former aims to design multi-class classification algorithms that can directly handle multi-class data, e.g., multi-class support vector machine [60] and softmax regression [3]. The latter aims to decompose multi-class classification into a set of binary classification problems via some rules, which is usually referred to as decomposition-based multi-class classification [30]. With the decomposed binary classification datasets, any off-the-shelf binary classification algorithms can be employed to learn binary classifiers. Then the final multi-class prediction for unseen instance can be determined via combining the predicted results of these binary classifiers. Compared with direct strategy, indirect strategy is more flexible as it can be coupled with any binary classification algorithms. Moreover, its effectiveness has also been validated via some empirical studies [21], [10].

For indirect strategy, the key to its success lies in how to decompose the original multi-class classification problem into binary classification problems and how to obtain the final prediction via combining the predicted results from binary classifiers [63]. These two processes are usually referred to as encoding and decoding, respectively. Specifically, the three most commonly used encoding strategies correspond to one-versus-one (OvO), one-versus-rest (OvR) and error-correcting output codes (ECOC) [9], where both the first two strategies can be regarded as special cases of the last one [2]. The basic decoding strategy for ECOC is Hamming decoding [9] which only utilizes binary predictions from binary classifiers. The loss-based decoding strategy [2] further considers the magnitude of predictions which usually indicates a level of "confidence". For both Hamming decoding and loss-based decoding, each binary classifier acts equally in the decoding process, while the generalization performance of different binary classifiers are usually different. To consider this difference, the loss-weighted decoding strategy [11] adjusts the importance of each binary classifier in the decoding process with their respective accuracies on each class.

However, existing studies have failed to consider the distinct impacts of decomposed binary classifiers on individual instances, potentially leading to performance degradation. Furthermore, the direct and indirect strategies in multi-class classification are often treated as two separate paths, without exploring their potential connections. To address these issues, this paper offers a more in-depth examination of the decoding process in decomposition-based multi-class classification. The main contributions of this paper are summarized as follows:

- We unify some common decoding methods under the

Bin-Bin Jia is with the College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China, and the Key Laboratory of New Generation Artificial Intelligence Technology & Its Interdisciplinary Applications (Southeast University), Ministry of Education, China. E-mail: jiabinbin@lut.edu.cn

Jun-Ying Liu is with the College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China. E-mail: liujy@lut.edu.cn

Min-Ling Zhang (corresponding author) is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. E-mail: zhangml@seu.edu.cn

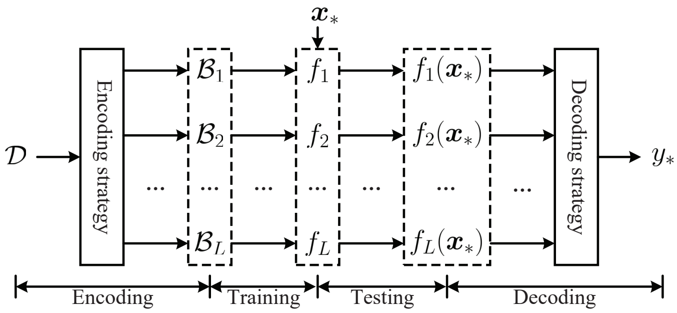


Fig. 1. An intuition for decomposition-based multi-class classification. Here, the multi-class dataset $\mathcal{D}$ is decomposed into L binary datasets $\mathcal{B}_1$, …, $\mathcal{B}_L$ according to the employed encoding strategy. Then, binary classifiers f_1, …, f_L are trained over the decomposed binary datasets $\mathcal{B}_1$, …, $\mathcal{B}_L$ with some binary classification algorithm (e.g., logistic regression), respectively. Finally, the final multi-class prediction y_* for unseen instance $\boldsymbol{x}_*$ is determined based on the predicted results $f_1(\boldsymbol{x}_*)$, …, $f_L(\boldsymbol{x}_*)$ according to the employed decoding strategy.

loss-based decoding strategy. Specifically, it is shown that the majority voting decoding in OvO is equivalent to Hamming decoding which is further shown as a special case of loss-based decoding with zero-one loss as loss function. It is also shown that determining the final class with maximum predicted confidence in OvR is also a special case of loss-based decoding with any monotone decreasing functions as loss function.

- We propose an instance-specific loss-weighted decoding strategy which allows for the differences in binary classifier's generalization ability for each sample. Specifically, the importance of each binary classifier for one specific sample is estimated with the classifier's accuracy in its neighboring samples. Experiments show the superiority of the proposed strategy over existing loss-based as well as loss-weighted decoding strategies.
- We argue that softmax regression is based on OvR decomposition, but just learns all decomposed binary classifiers in a joint manner. Then, we apply the proposed instance-specific loss-weighted decoding strategy in the decoding process of softmax regression. Experiments clearly show that the performance of softmax regression can be further improved with our decoding strategy.

The rest of this paper is organized as follows. Section II discusses related works on decomposition-based multi-class classification. Section III presents the preliminaries on ECOC and unifies some existing decoding strategies under loss-based decoding. Section IV presents the proposed instance-specific loss-weighted decoding strategy, including technical details and comparative studies. Section V presents how to apply the proposed decoding strategy in softmax regression and also reports the corresponding experimental results. Finally, we conclude this paper in Section VI.

II. Related Work

Generally, classification algorithms are initially designed for binary classification tasks. When we need to solve multi-class classification problems, some binary classification algorithms can be naturally generalized to deal with multi-class data, e.g., KNN classifier and naïve Bayes classifier [65], while some other binary classification algorithms need some special adaptions, e.g., logistic regression[1] and support vector machine [60]. Instead of fitting binary classification algorithms to multi-class data (i.e., direct strategy), another popular strategy is to fit multi-class data to binary classification algorithms (i.e., indirect strategy), which is termed as decomposition-based multi-class classification in this paper. Fig.1 shows an intuition of this strategy, which consists of four stages, including encoding, training, testing and decoding.

The encoding stage aims to decompose the original multi-class classification problem into a set of binary classification problems according to the designed rule. Intuitively, the difficulty of solving different binary classification problems is different. Thus, it will be better if the obtained binary classification problems are easier to solve. It is obvious that OvO and OvR will lead to fixed decomposed results, while the initial ECOC strategy randomly completes the decomposition process [9], [2]. All of them are problem-independent and cannot consider the specific characteristics in the original problem, which can lead to a decline in performance [6]. Regarding this issue, existing research mainly aims to design problem-dependent encoding strategy to obtain the most suitable decomposition rule for the specific multi-class classification problem [13]. For example, DECOC [50] hierarchically splits the class space into positive and negative parts with informatic metrics and SECOC [12] will further split some linearly non-separable classes into subclasses. In contrast to DECOC and SECOC, M2ECOC [75] and SM2ECOC [74] pick out positive and negative classes with maximum margin and then merge them together in next iteration. Different from SECOC, VL-ECOC [15] deals with hard class by longer codeword, leading to a variable-length ECOC algorithm. ECOC-ONE [49] aims to improve initial decompositions via iteratively creating new binary decompositions. In addition to these special designs, the encoding strategy is often optimized by using evolutionary algorithms [68], [32], [66].

Deep learning techniques usually deal with multi-class classification with the OvR-like one-hot encoding [52]. Therefore, it is natural to explore the combination of more general binary decomposition techniques with deep learning [48], [56], [59], [58]. One major disadvantage of the binary decomposition scheme is its limited ability to represent subtle between-class differences, prompting studies to explore N-ary decomposition, which aims to decompose the original multi-class classification problem into a set of simpler multi-class classification problems [77], [70], [45]. Moreover, recent studies also show that performance improves when classes are assigned subordination degrees instead of binary values (i.e., positive/negative class), initiating the study of soft-coded ECOC [40], [33], [39]. Besides, [20] specially aims to design scalable ECOC for large multi-class problems. After obtaining many decomposition results, D-Chooser provides an option to select the best one from them without a training stage [4].

[1] When logistic regression is generalized to solve multi-class classification problems, it is usually termed as softmax regression which is also known as multinomial logistic regression [3].

The training stage aims to learn binary classifiers based on the decomposed binary classification datasets. In general, these classifiers can be learned just like solving some independent normal binary classification problems. However, JCL [41] argued that correlations might exist among these decomposed binary classification problems [51] and the performance of these classifiers can be improved with consideration of potential correlations. On the other hand, different binary classification problems have their own special characteristics. If we can solve each problem based on a learning feature representation which is more discriminative for the problem than the original feature space, then better performance can be achieved [22]. The special characteristics can also be considered by learning binary classifiers with different binary classification algorithms [28].

The testing stage aims to obtain predictions of binary classifiers for unseen instance. It is usually studied together with the decoding stage, which aims to combine predictions of binary classifiers to obtain the final multi-class prediction [55]. Hamming decoding is the initial proposal to decode where Hamming distance is used to measure the similarity between binary predicted vector and the codeword of each class [9]. If Hamming distance is replaced with Euclidean distance, then Euclidean decoding is obtained [49]. A common disadvantage of these strategies is that the magnitude of the predictions is entirely ignored which can often indicate a level of "confidence". Loss-based decoding [2] and probabilistic-based decoding [47] utilize the magnitude of the predictions where the former computes the similarity with some loss functions (e.g., exponential loss) and the latter estimates the class probability with logistic function. Loss-weighted decoding [11] further considers generalization abilities of decomposed binary classifier to adjust their importance in the decoding process. However, some ensemble-based studies [43], [23] show that the generalization ability of one classifier might vary for different samples and the performance of ensemble-based model can be improved if the varying generalization ability can be properly utilized. This idea can be generalized to the decoding process in decomposition-based multi-class classification which aims at combining multiple binary classifiers to obtain a single multi-class classifier.

The idea of binary decomposition has also been applied to some learning problems related to multi-class classification. In partial label learning, PL-ECOC initiates the application of ECOC for partial label data in designing a disambiguation-free partial label method [72]. The following works mainly focus on improving the quality of encoding matrix [34], [54], [35] to achieve better binary decompositions. In multi-label learning, binary relevance [71] and calibrated label ranking [19] correspond to the application of OvR and OvO, respectively. However, most multi-label methods named after ECOC do not aim to decompose the original multi-label problem into several binary classification problems, but utilize the error correcting capability like noisy communication [27], [17]. RMSC, though named after ECOC, actually works in OvR mode [53]. In partial multi-label learning, PAMB adapts the ternary ECOC to enable binary decomposition [37] without an explicit disambiguation operation.

TABLE I
NOTATIONS.

Notations	Descriptions
m	the number of training samples
d	the number of features
N	the number of classes
L	the number of binary decompositions
$\mathcal{X}$	the d-dimensional feature space
$\mathcal{Y}$	the output space where $\mathcal{Y} = \{c_1, c_2, \ldots, c_N\}$
$\mathcal{D}$	the multi-class training set where $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$
$\mathcal{B}_l$	the binary dataset of the l-th binary decomposition
$\boldsymbol{x}_i$	the i-th d-dimensional feature vector where $\boldsymbol{x}_i \in \mathcal{X}$
y_i	the class label of $\boldsymbol{x}_i$ where $y_i \in \mathcal{Y}$
$\boldsymbol{x}_*$	the unseen instance where $\boldsymbol{x}_* \in \mathcal{X}$
y_*	the multi-class prediction for $\boldsymbol{x}_*$ where $y_* \in \mathcal{Y}$
$\mathbf{M}$	the $N \times L$ encoding matrix
M_{jl}	the (j, l)-th item of $\mathbf{M}$
$\mathbf{M}_{j:}$	the j-th row of $\mathbf{M}$
$\mathfrak{L}$	the employed binary classification algorithm
$f_l(\cdot)$	the binary classifier induced over $\mathcal{B}_l$
$f_l^b(\cdot)$	the specialized version of f_l returning binary prediction
$f_l^r(\cdot)$	the specialized version of f_l returning real-valued prediction
$f(\cdot)$	the concatenated binary classifiers: $f(\cdot) = [f_1(\cdot), \ldots, f_L(\cdot)]$
$\text{sign}(\cdot)$	the sign function where $f_l^b(\cdot) = \text{sign}(f_l^r(\cdot))$
$[\![\pi]\!]$	return 1 if condition π is true and 0 otherwise

III. PRELIMINARY

Let $\mathcal{X} = \mathbb{R}^d$ be the d-dimensional feature space and $\mathcal{Y} = \{c_1, c_2, \ldots, c_N\}$ be the output space with N classes, the task of multi-class classification is to learn a mapping from $\mathcal{X}$ to $\mathcal{Y}$ based on a set of training samples $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, where $\boldsymbol{x}_i \in \mathcal{X}$ is a d-dimensional feature vector and $y_i \in \mathcal{Y}$ is the corresponding class of $\boldsymbol{x}_i$. To facilitate understanding, Table I summarizes commonly-used notations in this paper.

A. Error-Correcting Output Codes

In ECOC, the decomposition rule can be denoted by a binary or ternary encoding matrix $\mathbf{M} \in \{-1, +1\}^{N \times L}$ or $\{-1, 0, +1\}^{N \times L}$, where the value of the (j, l)-th item M_{jl} in $\mathbf{M}$ indicates the role of samples belonging to the j-th class c_j in the l-th binary decomposition. Specifically, in the construction of the l-th binary dataset, a value of $M_{jl} = +1$ or -1 indicates that the class c_j is assigned as positive or negative. Moreover, a value of $M_{jl} = 0$ signifies the exclusion of the class c_j. For convenience, let $\mathcal{Y}_l^{+1}$, $\mathcal{Y}_l^{-1}$ and $\mathcal{Y}_l^{0}$ be the set of positive, negative and excluded classes in the l-th binary decomposition, they can be uniformly defined as follows:

$$\mathcal{Y}_l^s = \{c_j \mid M_{jl} = s, 1 \leq j \leq N\},\ (1 \leq l \leq L) \tag{1}$$

where $s \in \{-1, 0, +1\}$. The l-th binary decomposition corresponds to the following binary dataset $\mathcal{B}_l$:

$$\mathcal{B}_l = \mathcal{B}_l^+ \cup \mathcal{B}_l^- \tag{2}$$

Here, $\mathcal{B}_l^+$ and $\mathcal{B}_l^-$ denote the set of positive and negative samples in $\mathcal{B}_l$, which are respectively defined as follows:

$$\begin{aligned} \mathcal{B}_l^+ &= \{(\boldsymbol{x}_i, +1) \mid \forall (\boldsymbol{x}_i, y_i) \in \mathcal{D} \wedge y_i \in \mathcal{Y}_l^{+1}\} \\ \mathcal{B}_l^- &= \{(\boldsymbol{x}_i, -1) \mid \forall (\boldsymbol{x}_i, y_i) \in \mathcal{D} \wedge y_i \in \mathcal{Y}_l^{-1}\} \end{aligned} \tag{3}$$

Note that, according to the encoding rule of ternary ECOC, the sample $(\boldsymbol{x}_i, y_i)$ will be discarded when constructing $\mathcal{B}_l$ if $y_i \in \mathcal{Y}_l^0$ (i.e., excluded class set). With some binary classification algorithm $\mathfrak{L}$ (e.g., logistic regression), one binary classifier f_l can be induced over $\mathcal{B}_l$, i.e., $f_l = \mathfrak{L}(\mathcal{B}_l)$.

Given an unseen instance $\boldsymbol{x}_*$, each binary classifier f_l can return one prediction $f_l(\boldsymbol{x}_*)$. For convenience, we further use $f_l^b(\boldsymbol{x}_*) \in \{-1, +1\}$ and $f_l^r(\boldsymbol{x}_*) \in \mathbb{R}$ to denote the prediction if it is binary-valued and real-valued, respectively. The greater the value of $f_l^r(\boldsymbol{x}_*)$, the more likely $f_l^b(\boldsymbol{x}_*)$ is equal to $+1$. Generally, it is assumed that $f_l^b(\boldsymbol{x}_*) = \text{sign}(f_l^r(\boldsymbol{x}_*))$ where $\text{sign}(\cdot)$ denotes sign function. For binary classifiers which aim at learning a classification hyperplane $(\boldsymbol{w}, b)$: $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0$ to separate the two classes (e.g., logistic regression [29] and support vector machine [5]), this assumption naturally holds by simply setting $f_l^r(\boldsymbol{x}_*) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_* + b$.

To determine the final multi-class prediction y_* for $\boldsymbol{x}_*$, we need to combine the L predictions (a.k.a. decoding). The general form of decoding strategies can be written as follows:

$$y_* = c_{\hat{j}}, \text{ where } \hat{j} = \mathop{\arg\min}_{1 \le j \le N} \ \text{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) \tag{4}$$

where $f(\boldsymbol{x}_*) = [f_1(\boldsymbol{x}_*), f_2(\boldsymbol{x}_*), \ldots, f_L(\boldsymbol{x}_*)]$ denotes the prediction vector and $\mathbf{M}_{j:}$ denotes the j-th row of encoding matrix $\mathbf{M}$ (a.k.a. the codeword for the j-th class c_j). Different distance functions $\text{dist}(\cdot, \cdot)$ will correspond to different decoding strategies.

The most commonly used decoding strategy is *Hamming decoding*, where the distance function is defined as follows:

$$\text{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) = \sum_{l=1}^{L} |M_{jl}| \cdot [\![f_l^b(\boldsymbol{x}_*) \neq M_{jl}]\!] \tag{5}$$

Here, $[\![\pi]\!]$ returns 1 if condition π is true and 0 otherwise. The term $|M_{jl}|$ aims at excluding the influence of zeros in ternary encoding matrix $\mathbf{M}$ which is known as *attenuated decoding* [49]. It is easy to know that Eq.(5) calculates the Hamming distance between the binary prediction vector $f^b(\boldsymbol{x}_*) = [f_1^b(\boldsymbol{x}_*), f_2^b(\boldsymbol{x}_*), \ldots, f_L^b(\boldsymbol{x}_*)]$ and $\mathbf{M}_{j:}$.

Intuitively, the same prediction (e.g., positive class) might be obtained with different confidences. Hamming decoding simply treats the L binary predictions $f_l^b(\boldsymbol{x}_*)$ equally. To further consider the predicted confidence, loss-based decoding [2] utilizes the real-valued prediction in decoding process and defines the distance function as follows:

$$\text{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) = \sum_{l=1}^{L} |M_{jl}| \cdot \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) \tag{6}$$

In general, $\ell(\cdot)$ corresponds to some monotone nonincreasing functions. An intuition is that the larger the confidence $f_l^r(\boldsymbol{x}_*) \cdot M_{jl}$, the smaller the loss $\ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl})$. Here, the most commonly used loss function corresponds to the popular exponential loss (i.e., $\ell(z) = e^{-z}$) [2], [72], [37].

Loss-based decoding ignores one intuitive truth that the generalization performance of the L binary classifiers f_l is different. Generally, classifiers with better generalization performance should be given higher weight in the decoding process. Regarding this issue, loss-weighted decoding [11] attempts to weight the loss $\ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl})$ with the generalization performance of f_l. To evaluate the performance for each classifier f_l, the following performance matrix $\tilde{\mathbf{H}}$ is calculated with training set:

$$\tilde{H}_{jl} = \frac{\sum_{i=1}^{m} \left([\![f_l^b(\boldsymbol{x}_i) = M_{jl}]\!] \wedge [\![y_i = c_j]\!] \right)}{\sum_{i=1}^{m} [\![y_i = c_j]\!]} \tag{7}$$

It is easy to know that the (j, l)-th item $\tilde{H}_{jl}$ in $\tilde{\mathbf{H}}$ corresponds to the empirical accuracy of classifier f_l over the training samples belonging to the j-th class c_j. To make the total contribution of involved binary classifiers the same for each class in decoding process, $\tilde{\mathbf{H}}$ is further normalized by row to obtain the final weight matrix $\mathbf{H}$:

$$H_{jl} = \frac{\tilde{H}_{jl}}{\sum_{a=1}^{L} \tilde{H}_{ja}} \tag{8}$$

Note that $\tilde{H}_{jl} = 0$ if $M_{jl} = 0$ since $f_l^b(\boldsymbol{x}_i) \in \{-1, +1\}$ and then $[\![f_l^b(\boldsymbol{x}_i) = M_{jl}]\!]$ always returns 0. Therefore, no matter how many binary classifiers are involved in decoding for each class, their weights add up to 1. Loss-weighted decoding introduces the weight matrix $\mathbf{H}$ on the basis of loss-based decoding. Thus, the corresponding distance function is defined as follows:

$$\text{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) = \sum_{l=1}^{L} |M_{jl}| \cdot H_{jl} \cdot \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) \tag{9}$$

B. One-versus-One and Hamming Decoding

In OvO, there are a total of $\binom{N}{2}$ binary decompositions where one class is taken as positive and another one class is taken as negative for each binary decomposition. According to Eq.(2) and Eq.(3), suppose that the l-th binary dataset $\mathcal{B}_l$ is constructed with $\mathcal{Y}_l^{+1} = \{c_u\}$, $\mathcal{Y}_l^{-1} = \{c_v\}$ and $\mathcal{Y}_l^{0} = \mathcal{Y} \setminus \{c_u, c_v\}$ $(1 \le u \neq v \le N)$, then the learned binary classifier f_l over $\mathcal{B}_l$ will tell us whether one sample belongs to class c_u or c_v. For unseen instance $\boldsymbol{x}_*$, there are a total of $\binom{N}{2}$ votes returned by binary classifiers. Let s_{*j} be the number of votes for the j-th class c_j, its final multi-class prediction y_* is usually determined as follows:

$$y_* = c_{\hat{j}}, \text{ where } \hat{j} = \mathop{\arg\max}_{1 \le j \le N} \ s_{*j} \tag{10}$$

In other words, the class with most votes is returned as the final prediction (a.k.a. majority voting). As the most commonly used decision fusion strategy, some theoretical studies have analyzed the generalization bound of majority voting under various conditions, such as partially labeled training data [16] and an increasing number of independent voters [1].

For this decomposition rule, its encoding matrix $\mathbf{M}$ is a $N \times \binom{N}{2}$ ternary matrix. Each column has a '-1', a '$+1$' and $(N-2)$ '0's, and each row has a total of $N-1$ nonzero values ('-1' and '$+1$'). An example is shown in Eq.(11) for the OvO encoding matrix $(N = 4)$:

$$\mathbf{M} = \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix} \tag{11}$$

Theorem 1. *The majority voting decoding in Eq.(10) for OvO is equivalent to Hamming decoding for ECOC as defined by Eq.(4) and Eq.(5).*

Proof. It is easy to know that minimizing the distance in Eq.(5) is equivalent to maximizing the following distance:

$$\mathrm{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) = \sum_{l=1}^{L} |M_{jl}| \cdot [\![f_l^b(\boldsymbol{x}_*) = M_{jl}]\!] \tag{12}$$

Noting that the above distance exactly records the number of votes for the j-th class c_j and this completes the proof. □

Theorem 1 tells us that the commonly used majority voting decoding for OvO is just an equivalent implementation of Hamming decoding for general ECOC encoding strategy. In Section III-D, we will further show the relationship between Hamming decoding and loss-based decoding.

C. One-versus-Rest and Loss-based Decoding

In OvR, there are a total of N binary decompositions where one class is taken as positive and the rest classes are taken as negative for each binary decomposition. According to Eq.(2) and Eq.(3), the l-th binary dataset $\mathcal{B}_l$ is constructed with $\mathcal{Y}_l^{+1} = \{c_l\}$ and $\mathcal{Y}_l^{-1} = \mathcal{Y} \setminus c_l$ in OvR decomposition ($1 \leq l \leq N$). For unseen instance $\boldsymbol{x}_*$, its final multi-class prediction y_* is usually determined as follows:

$$y_* = c_{\hat{j}}, \text{ where } \hat{j} = \underset{1 \leq j \leq N}{\arg\max}\ f_j^r(\boldsymbol{x}_*) \tag{13}$$

In other words, the class with maximum confidence value is returned as the final prediction.

For this decomposition rule, its encoding matrix $\mathbf{M}$ is a $N \times N$ binary square matrix and can be written as $M_{jj} = +1$ and $M_{jl} = -1$ if $j \neq l$. An example is shown in Eq.(14) for the OvR encoding matrix ($N = 4$):

$$\mathbf{M} = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix} \tag{14}$$

Theorem 2. *The maximum confidence decoding in Eq.(13) for OvR is equivalent to loss-based decoding as defined by Eq.(4) and Eq.(6) if loss function $\ell(\cdot)$ is monotonically decreasing.*

Proof. For unseen instance $\boldsymbol{x}_*$, let $f_1^r(\boldsymbol{x}_*)$, $f_2^r(\boldsymbol{x}_*)$, …, $f_N^r(\boldsymbol{x}_*)$ be the N real-valued predictions, without loss of generality, suppose that Eq.(13) returns c_k as the final multi-class prediction, i.e., $f_k^r(\boldsymbol{x}_*)$ is the largest one among the N real-valued predictions, it is equivalent to proving that the following inequality always holds:

$$\sum_{l=1}^{N} \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{kl}) - \sum_{l=1}^{N} \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) < 0 \tag{15}$$

where $1 \leq j \neq k \leq N$. In other words, loss-based decoding will also return c_k as the final multi-class prediction.

In OvR, note that $M_{jl} = +1$ if $j = l$ and -1 otherwise. Thus, the two summations on the left side of Eq.(15) differ only in two terms (i.e., when $l = k$ and $l = j$):

$$\begin{aligned} &\sum_{l=1}^{N} \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{kl}) - \sum_{l=1}^{N} \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) \\ &= \ell(f_k^r(\boldsymbol{x}_*)) + \ell(-f_j^r(\boldsymbol{x}_*)) - \ell(-f_k^r(\boldsymbol{x}_*)) - \ell(f_j^r(\boldsymbol{x}_*)) \\ &= \big[\ell(f_k^r(\boldsymbol{x}_*)) - \ell(f_j^r(\boldsymbol{x}_*))\big] + \big[\ell(-f_j^r(\boldsymbol{x}_*)) - \ell(-f_k^r(\boldsymbol{x}_*))\big] \end{aligned}$$

As supposed before, $f_k^r(\boldsymbol{x}_*) > f_j^r(\boldsymbol{x}_*)$ hold, thus $-f_j^r(\boldsymbol{x}_*) > -f_k^r(\boldsymbol{x}_*)$ must hold. Because $\ell(z)$ is monotonically decreasing, then both $\ell(f_k^r(\boldsymbol{x}_*)) - \ell(f_j^r(\boldsymbol{x}_*)) < 0$ and $\ell(-f_j^r(\boldsymbol{x}_*)) - \ell(-f_k^r(\boldsymbol{x}_*)) < 0$ hold. Thus the inequality in Eq.(15) always holds and this completes the proof. □

Theorem 2 tells us that the commonly used maximum confidence decoding for OvR is just a special case of loss-based decoding for general ECOC. Thus, the performance of OvR-based models might be further improved with more advanced decoding strategy than loss-based decoding. Experiments in Sections IV-B and V-B clearly validate this conjecture.

It is worth noting that Hamming decoding is not suitable for OvR decomposition. Take the encoding matrix in Eq.(14) as an example, let $f^b(\boldsymbol{x}_*) = [f_1^b(\boldsymbol{x}_*), f_2^b(\boldsymbol{x}_*), f_3^b(\boldsymbol{x}_*), f_4^b(\boldsymbol{x}_*)]$ be the binary prediction vector, if there is one and only one '+1' in $f^b(\boldsymbol{x}_*)$, then the corresponding class can be returned by Hamming decoding (e.g., c_1 will be returned if $f^b(\boldsymbol{x}_*) = [+1, -1, -1, -1]$). However, two bad cases arise frequently in OvR decomposition. If there are multiple '+1's in $f^b(\boldsymbol{x}_*)$, the Hamming distances between $f^b(\boldsymbol{x}_*)$ and the corresponding rows in encoding matrix are equal as well as the nearest which will prevent picking out the proper class (e.g., if $f^b(\boldsymbol{x}_*) = [+1, +1, -1, -1]$, the Hamming distances between $f^b(\boldsymbol{x}_*)$ and both the first and second row in encoding matrix are equal to 1, and this is also the nearest distance. So which class should be returned?). Moreover, such similar problem also exists if there is no '+1' in $f^b(\boldsymbol{x}_*)$. This is why OvR uses maximum confidence decoding rather than Hamming decoding.

D. Hamming Decoding and Loss-based Decoding

Generally, Hamming decoding and loss-based decoding are treated as two separate decoding techniques. As discussed in Section III-A, loss-based decoding can be considered to be an improved version of Hamming decoding that can further regard the predicted confidence. In this section, we will demonstrate a more essential relationship between them.

Theorem 3. *When the loss function $\ell(\cdot)$ corresponds to zero-one loss $\ell_{0/1}(z)$, the distance function in Eq.(6) for loss-based decoding will be specialized into the distance function in Eq.(5) for Hamming decoding. Here, $\ell_{0/1}(z)$ returns* 1 *if* $z < 0$ *and* 0 *otherwise.*

Proof. As assumed in Section III-A, we have $f_l^b(\boldsymbol{x}_*) = \mathrm{sign}(f_l^r(\boldsymbol{x}_*))$. It is easy to show the following equation always holds for $M_{jl} \in \{-1, +1\}$ (no effect if $M_{jl} = 0$):

$$\ell_{0/1}(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) = [\![f_l^b(\boldsymbol{x}_*) \neq M_{jl}]\!] \tag{16}$$

Specifically, when $M_{jl} \in \{-1, +1\}$, if $f_l^b(\boldsymbol{x}_*) \neq M_{jl}$, i.e., $\text{sign}(f_l^r(\boldsymbol{x}_*)) \neq M_{jl}$, then $f_l^r(\boldsymbol{x}_*) \cdot M_{jl} < 0$ must hold, which leads to $\ell_{0/1}(f_l^r(\boldsymbol{x}_*) \cdot M_{jl})$ also returns 1. The proof will be completed by replacing the corresponding term in Eq.(5) or Eq.(6) with its equivalent form in Eq.(16). □

Theorem 3 tells us that the commonly used Hamming decoding for ECOC is just a special case of loss-based decoding. Thus, the performance of Hamming decoding might be further improved with proper weight matrix like loss-weighted decoding. Experiments in Section IV-B clearly validate this conjecture, especially our instance-specific loss-weighted decoding proposed in Section IV can significantly improve the performance of Hamming decoding.

E. Summary

In this section, we begin with a brief introduction to ECOC, with a special focus on its decoding strategy. We discuss the evolution from Hamming decoding to loss-based decoding, and to loss-weighted decoding. Subsequently, we prove the equivalence of majority voting in OvO with Hamming decoding, and that of maximum confidence decoding in OvR with loss-based decoding. Finally, we elucidate the relationship between Hamming decoding and loss-based decoding, demonstrating that the former can be considered a special case of the latter. These findings unify the commonly used decoding strategies in OvO, OvR, and ECOC under loss-based decoding. Therefore, these decoding strategies can be further enhanced by employing loss-weighted decoding with an appropriate weight matrix. In the next section, we will propose our instance-specific loss-weighted decoding strategy, which can consider the varying generalization ability of each binary classifier for each instance.

IV. Instance-specific Loss-weighted Decoding

In common experience, it is widely believed that different people are adept at different tasks. This also holds true for machine learning, where the generalization ability of one classifier may vary when it is used to classify different samples [43], [23]. Given this fundamental insight, in decomposition-based multi-class classification, its performance could potentially be enhanced by assigning sample-dependent weights to each binary classifier during the decoding process, provided that these weights can reflect each classifier's generalization ability for each sample. The loss-weighted decoding strategy assigns the same weight to each binary classifier for all possible samples. Thus, it fails to consider the individual generalization ability of each binary classifier for each sample, which may result in performance degradation. In this section, we propose to estimate the distinct instance-specific importance for each binary classifier during the decoding of the final multi-class prediction for each sample.

A. Methodology

Following the notations in previous sections, for convenience, we rewrite the dataset for the l-th binary decomposition in Eq.(2) as follows:

$$\mathcal{B}_l = \{(\boldsymbol{x}_i^l, y_i^l) \mid 1 \leq i \leq |\mathcal{B}_l|\},\ (1 \leq l \leq L) \tag{17}$$

where $\boldsymbol{x}_i^l \in \mathcal{X}$, $y_i^l \in \{-1, +1\}$ and $|\cdot|$ returns the cardinality of one set. It is easy to know $|\mathcal{B}_l|$ is equal to the number of training samples whose class is in $\mathcal{Y}_l^{+1} \cup \mathcal{Y}_l^{-1}$ (cf. Eq.(1) for their definitions). For the unseen instance $\boldsymbol{x}_*$, we firstly identify its K nearest neighbors in the l-th binary dataset $\mathcal{B}_l$:

$$\mathcal{N}^l(\boldsymbol{x}_*) = \{(\boldsymbol{x}_{i_k}^l, y_{i_k}^l) \mid 1 \leq k \leq K\} \tag{18}$$

Generally, the neighboring samples have similar properties as $\boldsymbol{x}_*$. Thus, the accuracy of f_l over $\mathcal{N}^l(\boldsymbol{x}_*)$ can reflect its specific generalization ability for $\boldsymbol{x}_*$. Specifically, for each neighboring sample $\boldsymbol{x}_{i_k}^l$, f_l can return a binary prediction $f_l^b(\boldsymbol{x}_{i_k}^l)$. Then, the local accuracy of f_l in neighboring $\boldsymbol{x}_*$ can be calculated as follows:

$$\tilde{H}_l^{\boldsymbol{x}_*} = \frac{1}{K}\sum_{k=1}^{K} [\![f_l^b(\boldsymbol{x}_{i_k}^l) = y_{i_k}^l]\!] \tag{19}$$

Here, $[\![\pi]\!]$ returns 1 if condition π is true and 0 otherwise. After all L accuracies are obtained, similar to Eq.(8), we can further normalize each accuracy as follows:

$$H_l^{\boldsymbol{x}_*} = \frac{\tilde{H}_l^{\boldsymbol{x}_*}}{\sum_{a=1}^{L} \tilde{H}_a^{\boldsymbol{x}_*}} \tag{20}$$

With the normalized accuracies, the corresponding distance function in Eq.(4) for instance-specific loss-weighted decoding can be finally defined as follows:

$$\text{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:}) = \sum_{l=1}^{L} |M_{jl}| \cdot H_l^{\boldsymbol{x}_*} \cdot \ell(f_l^r(\boldsymbol{x}_*) \cdot M_{jl}) \tag{21}$$

Note that the weight $H_l^{\boldsymbol{x}_*}$ is instance-dependent and can be regarded as the generalization ability estimation of f_l for $\boldsymbol{x}_*$, thus this decoding strategy can consider instance-specific characteristics of each binary for each instance.

To facilitate understanding, Algorithm 1 presents the pseudocode of instance-specific loss-weighted decoding.

B. Experiments

TABLE II
The characteristics of multi-class datasets.

Dataset	#Sample (m)	#Class (N)	#Feature (d)
iris	150	3	4
wine	178	3	13
glass	214	6	9
svmguide2	391	3	20
vowel	528	11	10
dna	2000	3	180
segment	2310	7	19
satimage	4435	6	36
usps	7291	10	256
pendigits	7494	10	16
letter	15000	26	16
protein	17766	3	357
poker	25010	10	10
shuttle	43500	7	9
mnist	60000	10	780

Algorithm 1 Instance-specific loss-weighted decoding.

```
Input: The $N \times L$ encoding matrix $\mathbf{M}$, the $L$ decomposed binary classification datasets $\mathcal{B}_l$s, the $L$ learned binary classifiers $f_l$s, the number of nearest neighbors $K$, and the unseen instance $\boldsymbol{x}_*$
Output: The final multi-class prediction $y_*$ for $\boldsymbol{x}_*$
for $l = 1$ to $L$ do
    Identify the $K$ nearest neighbors of $\boldsymbol{x}_*$ in $\mathcal{B}_l$ and store them in sample set $\mathcal{N}^l(\boldsymbol{x}_*)$ as shown in Eq.(18);
    Calculate the local accuracy $\tilde{H}_l^{\boldsymbol{x}_*}$ of $f_l$ in $K$ nearest neighbors $\mathcal{N}^l(\boldsymbol{x}_*)$ according to Eq.(19);
end for
for $l = 1$ to $L$ do
    Calculate the normalized accuracy $H_l^{\boldsymbol{x}_*}$ according to Eq.(20);
end for
for $j = 1$ to $N$ do
    Calculate the distance $\mathrm{dist}(f(\boldsymbol{x}_*), \mathbf{M}_{j:})$ according to Eq.(21);
end for
Return $y_*$ according to Eq.(4).
```

TABLE III
EXPERIMENTAL RESULTS (MEAN±STD.) FOR BINARY ECOC WHERE THE BEST AND SECOND PERFORMANCE FOR EACH DATASET IS HIGHLIGHTED IN BOLDFACE AND UNDERLINED, RESPECTIVELY.

(a) *Accuracy*

Dataset	LB-0/1	LW-0/1	ILW-0/1	LB-Exp	LW-Exp	ILW-Exp
iris	0.720±0.069	**0.953±0.055**	**0.953±0.055**	0.913±0.095	0.940±0.073	0.947±0.061
wine	0.972±0.040	0.972±0.040	0.972±0.040	**0.983±0.027**	**0.983±0.027**	**0.983±0.027**
glass	0.542±0.100	0.547±0.120	0.580±0.149	0.576±0.088	**0.627±0.121**	0.618±0.127
svmguide2	0.796±0.053	0.806±0.043	0.826±0.055	**0.834±0.058**	0.829±0.057	0.831±0.059
vowel	0.394±0.058	0.479±0.061	**0.680±0.045**	0.451±0.101	0.517±0.056	0.612±0.085
dna	0.912±0.023	0.930±0.016	0.916±0.021	**0.944±0.021**	**0.944±0.021**	**0.944±0.020**
segment	0.880±0.029	0.907±0.019	**0.935±0.018**	0.906±0.017	0.913±0.017	0.929±0.017
satimage	0.804±0.021	0.846±0.023	0.875±0.017	0.835±0.017	0.855±0.016	**0.876±0.014**
usps	0.916±0.011	0.920±0.011	0.930±0.012	0.933±0.010	0.934±0.011	**0.939±0.012**
pendigits	0.868±0.016	0.882±0.015	**0.973±0.005**	0.901±0.011	0.903±0.011	0.968±0.006
letter	0.429±0.013	0.552±0.017	**0.856±0.014**	0.467±0.022	0.583±0.021	0.843±0.015
protein	0.669±0.013	0.642±0.013	0.665±0.011	**0.684±0.014**	0.683±0.015	**0.684±0.013**
poker	0.498±0.011	0.301±0.019	**0.502±0.010**	0.499±0.010	0.424±0.012	0.500±0.011
shuttle	0.834±0.064	0.956±0.002	**0.975±0.003**	0.924±0.004	0.952±0.004	**0.975±0.003**
mnist	0.855±0.011	0.860±0.010	0.895±0.008	0.874±0.008	0.876±0.008	**0.898±0.006**

(b) *Average*-F1

Dataset	LB-0/1	LW-0/1	ILW-0/1	LB-Exp	LW-Exp	ILW-Exp
iris	0.693±0.066	**0.951±0.059**	**0.951±0.059**	0.913±0.089	0.937±0.073	0.942±0.067
wine	0.969±0.042	0.969±0.042	0.969±0.042	**0.977±0.037**	**0.977±0.037**	**0.977±0.037**
glass	0.436±0.133	0.505±0.124	0.471±0.188	0.480±0.105	**0.573±0.145**	0.525±0.164
svmguide2	0.664±0.103	0.708±0.080	**0.753±0.093**	0.743±0.118	0.737±0.121	0.739±0.123
vowel	0.347±0.068	0.452±0.060	**0.651±0.056**	0.407±0.108	0.492±0.063	0.582±0.100
dna	0.903±0.023	0.923±0.018	0.907±0.023	**0.935±0.023**	**0.935±0.023**	**0.935±0.022**
segment	0.873±0.029	0.906±0.017	**0.934±0.015**	0.904±0.016	0.912±0.015	0.927±0.016
satimage	0.683±0.022	0.793±0.026	0.828±0.017	0.737±0.016	0.804±0.021	**0.831±0.010**
usps	0.907±0.012	0.911±0.013	0.922±0.014	0.925±0.012	0.926±0.012	**0.932±0.014**
pendigits	0.865±0.016	0.880±0.015	**0.972±0.005**	0.899±0.011	0.901±0.012	0.968±0.006
letter	0.393±0.010	0.535±0.020	**0.855±0.013**	0.425±0.022	0.564±0.023	0.843±0.013
protein	0.632±0.015	0.633±0.013	0.643±0.013	**0.664±0.015**	**0.664±0.015**	**0.664±0.014**
poker	0.083±0.007	0.078±0.008	**0.088±0.008**	0.082±0.008	0.073±0.007	0.082±0.008
shuttle	0.344±0.147	0.569±0.078	**0.715±0.071**	0.494±0.077	0.536±0.083	0.677±0.073
mnist	0.852±0.011	0.858±0.010	0.893±0.008	0.872±0.008	0.874±0.008	**0.897±0.006**

1) Experimental Setup: In this paper, we have collected fifteen publicly available multi-class datasets for comparative studies[2]. Table II summarizes the characteristics of these datasets, including the number of samples (i.e., m), the number of classes (i.e., N) and the number of features (i.e., d).

To measure the performance of different multi-class classification models, we use the two popular evaluation metrics including accuracy and average-F1 [38], [22]. Specifically, given the multi-class test set $\mathcal{S} = \{(\boldsymbol{x}_i, y_i) \mid 1 \leq i \leq p\}$ with p samples where $y_i \in \{c_1, c_2, \ldots, c_N\}$, for the multi-class classifier f to be evaluated, let $\hat{y}_i = f(\boldsymbol{x}_i)$ be the multi-

[2] These datasets are publicly available at https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html

TABLE IV
EXPERIMENTAL RESULTS (MEAN±STD.) FOR TERNARY ECOC WHERE THE BEST AND SECOND PERFORMANCE FOR EACH DATASET IS HIGHLIGHTED IN BOLDFACE AND UNDERLINED, RESPECTIVELY.

(a) *Accuracy*

Dataset	LB-0/1	LW-0/1	ILW-0/1	LB-Exp	LW-Exp	ILW-Exp
iris	**0.953±0.055**	**0.953±0.055**	**0.953±0.055**	**0.953±0.055**	**0.953±0.055**	**0.953±0.055**
wine	**0.989±0.024**	**0.989±0.024**	**0.989±0.024**	0.983±0.027	0.983±0.027	0.983±0.027
glass	0.575±0.108	0.523±0.105	**0.594±0.117**	**0.594±0.129**	0.552±0.115	**0.594±0.124**
svmguide2	0.824±0.041	0.826±0.046	**0.829±0.047**	0.826±0.050	0.826±0.050	**0.829±0.053**
vowel	0.481±0.078	0.557±0.065	0.650±0.053	0.557±0.091	0.585±0.072	**0.655±0.058**
dna	0.941±0.015	0.940±0.015	0.941±0.015	**0.945±0.018**	**0.945±0.018**	**0.945±0.018**
segment	0.914±0.024	0.919±0.022	**0.933±0.018**	0.912±0.021	0.918±0.022	0.931±0.014
satimage	0.842±0.019	0.855±0.021	**0.882±0.019**	0.851±0.019	0.860±0.019	0.878±0.022
usps	0.949±0.007	0.949±0.008	0.952±0.008	0.952±0.010	0.953±0.010	**0.954±0.010**
pendigits	0.936±0.008	0.937±0.010	0.971±0.009	0.946±0.007	0.946±0.007	**0.972±0.005**
letter	0.538±0.011	0.601±0.013	**0.834±0.016**	0.594±0.018	0.643±0.022	0.831±0.012
protein	0.682±0.014	0.683±0.014	0.682±0.014	0.684±0.014	0.683±0.015	**0.685±0.015**
poker	0.499±0.010	0.386±0.079	0.500±0.010	0.500±0.010	0.402±0.055	**0.501±0.013**
shuttle	0.942±0.034	0.964±0.001	0.979±0.002	0.940±0.009	0.964±0.003	**0.982±0.007**
mnist	0.900±0.005	0.903±0.006	0.914±0.004	0.911±0.005	0.911±0.005	**0.919±0.004**

(b) *Average*-F1

Dataset	LB-0/1	LW-0/1	ILW-0/1	LB-Exp	LW-Exp	ILW-Exp
iris	**0.951±0.059**	**0.951±0.059**	**0.951±0.059**	**0.951±0.059**	**0.951±0.059**	**0.951±0.059**
wine	**0.985±0.032**	**0.985±0.032**	**0.985±0.032**	0.977±0.037	0.977±0.037	0.977±0.037
glass	0.510±0.157	0.520±0.136	**0.539±0.152**	0.526±0.162	0.538±0.140	0.525±0.156
svmguide2	0.740±0.105	0.741±0.109	**0.754±0.092**	0.740±0.117	0.740±0.118	0.741±0.122
vowel	0.448±0.089	0.541±0.079	0.624±0.075	0.537±0.111	0.564±0.090	**0.647±0.057**
dna	0.931±0.018	0.932±0.018	0.931±0.018	**0.937±0.020**	**0.937±0.020**	**0.937±0.020**
segment	0.913±0.022	0.919±0.021	**0.933±0.015**	0.912±0.021	0.917±0.022	0.930±0.012
satimage	0.771±0.042	0.812±0.028	**0.844±0.024**	0.783±0.028	0.818±0.021	0.835±0.027
usps	0.943±0.008	0.943±0.009	0.946±0.009	0.947±0.011	0.947±0.011	**0.949±0.011**
pendigits	0.936±0.008	0.936±0.010	0.970±0.009	0.945±0.006	0.946±0.007	**0.972±0.005**
letter	0.511±0.014	0.589±0.014	**0.830±0.018**	0.570±0.023	0.630±0.025	0.829±0.012
protein	0.662±0.016	0.664±0.016	0.663±0.016	0.664±0.015	**0.665±0.016**	**0.665±0.016**
poker	0.083±0.009	0.080±0.013	0.084±0.009	0.082±0.008	0.072±0.008	**0.086±0.011**
shuttle	0.546±0.094	0.642±0.081	**0.700±0.065**	0.543±0.077	0.592±0.070	0.687±0.077
mnist	0.898±0.005	0.901±0.006	0.913±0.004	0.910±0.005	0.910±0.005	**0.918±0.004**

TABLE V
WILCOXON SIGNED-RANKS TEST FOR BINARY ECOC (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
LW-0/1 vs.		
LB-0/1	**win**[4.19e-02] (8/2/0)	**win**[8.54e-04](10/0/0)
ILW-0/1 vs.		
LB-0/1	**win**[6.10e-04](10/0/0)	**win**[1.22e-04](10/0/0)
LW-0/1	**win**[7.32e-04](10/0/0)	**win**[1.05e-02](10/0/0)
LB-Exp vs.		
LB-0/1	**win**[6.10e-05](10/0/0)	**win**[1.22e-04](10/0/0)
LW-0/1	**tie**[5.61e-01](0/10/0)	**tie**[4.54e-01](0/10/0)
ILW-0/1	**loss**[4.79e-02](0/0/10)	**loss**[3.02e-02](0/0/10)
LW-Exp vs.		
LB-0/1	**win**[2.62e-03](10/0/0)	**win**[1.83e-04](10/0/0)
LW-0/1	**win**[8.54e-04](10/0/0)	**win**[2.15e-02](10/0/0)
ILW-0/1	**tie**[1.21e-01] (0/1/9)	**tie**[1.21e-01] (0/6/4)
LB-Exp	**win**[8.03e-02](10/0/0)	**win**[1.71e-02](10/0/0)
ILW-Exp vs.		
LB-0/1	**win**[6.10e-05](10/0/0)	**win**[1.22e-04](10/0/0)
LW-0/1	**win**[1.22e-04](10/0/0)	**win**[3.05e-04](10/0/0)
ILW-0/1	**tie**[5.99e-01](0/10/0)	**tie**[7.62e-01](0/10/0)
LB-Exp	**win**[4.03e-03](10/0/0)	**win**[3.05e-03](10/0/0)
LW-Exp	**win**[2.32e-03](10/0/0)	**win**[1.07e-02](10/0/0)

TABLE VI
WILCOXON SIGNED-RANKS TEST FOR TERNARY ECOC (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
LW-0/1 vs.		
LB-0/1	**tie**[2.04e-01] (3/7/0)	**win**[4.64e-03](10/0/0)
ILW-0/1 vs.		
LB-0/1	**win**[4.88e-04](10/0/0)	**win**[2.44e-04](10/0/0)
LW-0/1	**win**[9.77e-04](10/0/0)	**win**[1.22e-03](10/0/0)
LB-Exp vs.		
LB-0/1	**win**[1.34e-02](10/0/0)	**win**[2.45e-02] (8/2/0)
LW-0/1	**tie**[5.42e-01](0/10/0)	**tie**[5.83e-01](0/10/0)
ILW-0/1	**loss**[8.54e-03](0/0/10)	**loss**[3.05e-03](0/0/10)
LW-Exp vs.		
LB-0/1	**tie**[1.53e-01] (6/4/0)	**win**[1.34e-02](10/0/0)
LW-0/1	**win**[8.54e-03] (5/5/0)	**tie**[2.68e-01](0/10/0)
ILW-0/1	**loss**[3.05e-03](0/0/10)	**loss**[5.25e-03](0/0/10)
LB-Exp	**tie**[4.65e-01](0/10/0)	**win**[2.69e-02](9/1/0)
ILW-Exp vs.		
LB-0/1	**win**[1.71e-03](10/0/0)	**win**[1.71e-03](10/0/0)
LW-0/1	**win**[1.22e-03](10/0/0)	**win**[3.05e-03](10/0/0)
ILW-0/1	**tie**[4.26e-01](0/10/0)	**tie**[5.83e-01](0/10/0)
LB-Exp	**win**[9.77e-04](10/0/0)	**win**[2.44e-03](10/0/0)
LW-Exp	**win**[4.88e-04](10/0/0)	**win**[6.84e-03](10/0/0)

TABLE VII
WILCOXON SIGNED-RANKS TEST FOR OVR (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
LW-0/1 vs.		
LB-0/1	**tie**[1.04e-01]	**win**[1.22e-04]
ILW-0/1 vs.		
LB-0/1	**win**[3.66e-04]	**win**[1.22e-04]
LW-0/1	**win**[7.32e-04]	**win**[2.44e-03]
LB-Exp vs.		
LB-0/1	**win**[1.22e-04]	**win**[1.22e-04]
LW-0/1	**win**[2.01e-03]	**win**[2.56e-02]
ILW-0/1	**tie**[1.07e-01]	**tie**[1.07e-01]
LW-Exp vs.		
LB-0/1	**win**[1.53e-03]	**win**[1.83e-04]
LW-0/1	**win**[4.27e-03]	**win**[8.36e-03]
ILW-0/1	**loss**[7.30e-02]	**tie**[1.69e-01]
LB-Exp	**loss**[6.81e-02]	**tie**[6.35e-01]
ILW-Exp vs.		
LB-0/1	**win**[6.10e-05]	**win**[6.10e-05]
LW-0/1	**win**[1.22e-04]	**win**[2.62e-03]
ILW-0/1	**tie**[9.34e-01]	**tie**[3.89e-01]
LB-Exp	**win**[5.25e-03]	**win**[4.03e-03]
LW-Exp	**win**[3.66e-04]	**win**[1.22e-03]

TABLE VIII
WILCOXON SIGNED-RANKS TEST FOR OVO (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
LW-0/1 vs.		
LB-0/1	**tie**[5.70e-01]	**win**[5.22e-02]
ILW-0/1 vs.		
LB-0/1	**win**[3.71e-02]	**win**[2.73e-02]
LW-0/1	**tie**[1.02e-01]	**tie**[3.39e-01]
LB-Exp vs.		
LB-0/1	**win**[7.71e-02]	**win**[3.27e-02]
LW-0/1	**tie**[1.29e-01]	**tie**[3.76e-01]
ILW-0/1	**tie**[7.91e-01]	**tie**[1.00e+00]
LW-Exp vs.		
LB-0/1	**tie**[6.26e-01]	**win**[7.85e-02]
LW-0/1	**tie**[6.70e-01]	**tie**[1.53e-01]
ILW-0/1	**tie**[3.76e-01]	**tie**[8.08e-01]
LB-Exp	**tie**[7.42e-01]	**tie**[1.64e-01]
ILW-Exp vs.		
LB-0/1	**win**[4.25e-02]	**win**[2.66e-02]
LW-0/1	**win**[3.42e-02]	**win**[1.71e-02]
ILW-0/1	**tie**[1.76e-01]	**tie**[4.14e-01]
LB-Exp	**tie**[1.48e-01]	**win**[7.42e-02]
LW-Exp	**win**[1.37e-02]	**tie**[3.22e-01]

class prediction for test sample $\boldsymbol{x}_i$, then the two metrics can be defined as follows:

- *Accuracy*:

$$Acc_{\mathcal{S}}(f) = \frac{1}{p}\sum_{i=1}^{p}[\![\hat{y}_i = y_i]\!] \tag{22}$$

Here, $[\![\pi]\!]$ returns 1 if condition π is true and 0 otherwise.

- *Average*-F1:

$$AvgF1_{\mathcal{S}}(f) = \frac{1}{N}\sum_{j=1}^{N}\frac{2P_j \cdot R_j}{P_j + R_j} \tag{23}$$

TABLE IX
WILCOXON SIGNED-RANKS TEST FOR GEPECOC (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
LW-0/1 vs.		
LB-0/1	**tie**[1.51e-01]	**win**[6.84e-03]
ILW-0/1 vs.		
LB-0/1	**win**[4.88e-04]	**win**[4.88e-04]
LW-0/1	**win**[6.84e-03]	**win**[6.40e-02]
LB-Exp vs.		
LB-0/1	**win**[3.27e-02]	**win**[1.71e-03]
LW-0/1	**tie**[1.76e-01]	**tie**[4.97e-01]
ILW-0/1	**tie**[4.97e-01]	**tie**[2.16e-01]
LW-Exp vs.		
LB-0/1	**tie**[4.14e-01]	**win**[1.34e-02]
LW-0/1	**tie**[7.65e-01]	**win**[9.42e-02]
ILW-0/1	**tie**[1.91e-01]	**tie**[5.88e-01]
LB-Exp	**tie**[6.25e-01]	**tie**[2.40e-01]
ILW-Exp vs.		
LB-0/1	**win**[5.74e-02]	**win**[7.32e-04]
LW-0/1	**win**[5.22e-02]	**win**[1.71e-02]
ILW-0/1	**tie**[3.05e-01]	**win**[8.03e-02]
LB-Exp	**win**[3.71e-02]	**win**[2.93e-03]
LW-Exp	**win**[1.95e-03]	**win**[5.37e-02]

Here, the definitions of P_j and R_j are given as follows:

$$P_j = \frac{\sum_{i=1}^{p}[\![\hat{y}_i = c_j]\!] \wedge [\![y_i = c_j]\!]}{\sum_{i=1}^{p}[\![\hat{y}_i = c_j]\!]}$$
$$R_j = \frac{\sum_{i=1}^{p}[\![\hat{y}_i = c_j]\!] \wedge [\![y_i = c_j]\!]}{\sum_{i=1}^{p}[\![y_i = c_j]\!]}$$

For both the two metrics, it is easy to know that the larger the metric value, the better the performance. We conduct ten-fold cross validation over each dataset and report both mean and standard deviation in experiments.

We compare the proposed instance-specific loss-weighted decoding strategy (abbreviated as ILW) with existing strategies including loss-based decoding (abbreviated as LB) [2] and loss-weighted decoding (abbreviated as LW) [11]. For the loss function $\ell(\cdot)$, we employ the popular exponential loss which has been widely used in diverse applications [72], [37]. Moreover, as discussed in Section III-D, Hamming decoding is a special case of loss-based decoding when zero-one loss serves as the loss function. Therefore, in addition to exponential loss, we also investigate zero-one loss in experiments. For convenience, let $\mathcal{A} \in \{\text{LB, LW, ILW}\}$ be one decoding strategy, we use $\mathcal{A}$-0/1 and $\mathcal{A}$-Exp to denote the version coupled with zero-one loss and exponential loss, respectively. For example, LB-0/1 denotes the loss-based decoding with loss function being zero-one loss (i.e., Hamming decoding).

For encoding strategy, we investigate the two random ECOC approaches (i.e., binary ECOC and ternary ECOC), the two deterministic ECOC approaches (i.e., OvR and OvO) and one state-of-the-art ECOC approach GEPECOC [66]. For the first four ECOC versions, their corresponding encoding matrices are generated by the built-in function `designecoc` in Matlab with parameter setting ‘*denserandom*’, ‘*sparserandom*’, ‘*onevsall*’ and ‘*onevsone*’, respectively. For GEPECOC, its corresponding encoding matrix is generated with the released code by authors.[3] For decomposed binary classification problems, logistic regression is used to learn binary classifier which is implemented by the efficient LIBLINEAR software [14] with parameter setting '-s 0 -B 1 -R -q', i.e., "L2-regularized logistic regression (primal)".

2) Experimental Results: Due to page limit, we only report the detailed experimental results for binary ECOC and ternary ECOC in Table III and Table IV, respectively. As the relative performance of different approaches might vary over different datasets due to their specific data distribution, in this paper, in order to obtain more robust experimental conclusion, we pay more attention to compare the overall performance of each approach over the whole benchmark datasets. To achieve this goal, Wilcoxon signed-ranks test [8] is employed as the statistical tool. Tables V-IX summarize the corresponding test results for binary ECOC, ternary ECOC, OvR, OvO and GEPECOC[4], respectively. The win/tie/loss in each cell for "A vs. B" means that the decoding strategy A achieves superior/comparable/inferior performance against the decoding strategy B. Take the binary ECOC in Table V as an example, the test result in bottom right corner (i.e., **win**[1.07e-02]) means that ILW-Exp achieves superior performance against LW-Exp and this is also equivalent to that LW-Exp achieves inferior performance against ILW-Exp. Besides, to investigate the impact of randomness in generating binary ECOC and ternary ECOC, we repeatedly run their programs ten times with different random seeds. In Table V and Table VI, we also show the win/tie/loss counts of the ten experiments for each setting. The detailed experimental results in Table III and Table IV and the statistical test results in Table V and Table VI only correspond to one of the ten experiments.

According to the reported experimental results, we can make the following observations:

- As shown in Tables V-IX, ILW achieves superior performance against LB and LW in most cases with either encoding strategies (binary ECOC, ternary ECOC, OvR, OvO and GEPECOC) as well as loss functions (zero-one loss or exponential loss). These experimental results not only validate the effectiveness of instance-specific loss-weighted decoding but also support our previous claim that neglecting the distinct impacts of the decomposed binary classifiers on individual instances can potentially lead to performance degradation.
- It is also shown that Hamming decoding (i.e., LB-0/1) achieves inferior performance to other decoding strategies in most cases, especially to the proposed ILW strategy. Hamming decoding was proposed along with ECOC, and it is the most commonly-used decoding strategy. These experimental results show that its performance can be further improved by considering the instance-specific characteristics of binary classifiers.
- It can be observed that there are some differences between ILW-0/1 and ILW-Exp in detailed experimental results, while they achieve comparable performance according to Wilcoxon signed-ranks test results. Different multi-class classification applications may prefer to different loss functions and it is encouraged to select the loss function according to the task in practice.
- As shown in Table III and Table IV, the relative performance of different decoding strategies is slightly different on different datasets. Specifically, we may expect that the performance rank is "ILW≥LW≥LB" and "LB-Exp≥LB-0/1 (i.e., Hamming decoding)", but this is not always true over all datasets. Note that, the "LB-Exp≥LB-0/1" holds when the quality of predicted confidence is reliable, "LW≥LB" holds when the quality of estimated weight matrix is reliable, and "ILW≥LW" holds when the quality of estimated weight matrix in neighboring samples is more reliable. However, these reliabilities should be examined in different real-world applications that depend on specific data distribution.
- As shown in Table V and Table VI, in the ten experiments, the statistical test result for a single setting keeps unchanged in most cases, i.e., win/tie/loss counts are either ten wins or ten ties or ten losses. Especially for the proposed ILW-Exp strategy, the statistical test result keeps unchanged in all cases. These experimental results demonstrate that the randomness in encoding matrix has little impacts on the final results.

TABLE X
WILCOXON SIGNED-RANKS TEST FOR ABLATION STUDY (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

(a) Loss function: zero-one loss

	Accuracy	*Average*-F1
Binary ECOC	**win**[1.22e-04]	**win**[6.10e-05]
Ternary ECOC	**win**[1.22e-04]	**win**[1.22e-04]
OvR	**win**[1.22e-04]	**win**[6.10e-05]
OvO	**win**[1.37e-02]	**win**[1.86e-02]
GEPECOC	**win**[4.88e-04]	**win**[4.88e-04]

(b) Loss function: exponential loss

	Accuracy	*Average*-F1
Binary ECOC	**win**[4.03e-03]	**win**[3.05e-03]
Ternary ECOC	**win**[3.05e-03]	**win**[8.54e-03]
OvR	**win**[5.25e-03]	**win**[4.03e-03]
OvO	**win**[4.20e-02]	**win**[2.69e-02]
GEPECOC	**win**[1.37e-02]	**win**[2.10e-02]

3) Ablation Study: In this paper, the proposed ILW strategy estimates the generalization ability of one binary classifier on one sample with its accuracy in the sample's k nearest neighbors. It is easy to understand that this accuracy and the prediction confidence on this sample are likely to be highly correlated. Then we can also consider simply using the prediction confidence of each binary classifier as its weight. To make real-valued prediction confidence being in [0, 1] that is similar to the value of accuracy, we normalize each real-valued prediction confidence $f_l^r(\boldsymbol{x})$ with sigmoid function $s(z) = \frac{1}{1+\exp(-z)}$. As the magnitude of prediction confidence indicates the possibility of belonging to positive class for one sample, the weight is set to $s(f_l^r(\boldsymbol{x}))$ for one positive class while $1 - s(f_l^r(\boldsymbol{x}))$ for one negative class.

[3] The code of GEPECOC is publicly available at https://github.com/MLDMXM2017/GEP_ECOC.

[4] As the encoding matrices of GEPECOC for datasets "letter" and "mnist" are not returned in one week, the reported statistical results in Table IX only employ the remaining 13 datasets.

Due to page limit, the detailed experimental results of this ablation version is omitted in this paper. Here, we pay more attention to compare the overall performance of ILW against this ablation version over the whole benchmark datasets. Table X reports the Wilcoxon signed-ranks test results for all the five ECOC versions and the two loss functions. It can be observed that ILW achieves superior performance against this ablation version in all cases. These experimental results show that the reliability of the estimated accuracy in neighboring samples is better than the prediction confidence.

4) Parameter Sensitivity Analysis: To estimate the instance-specific generalization ability of one decomposed binary classifier, the accuracy in K nearest neighbors of one instance serves as this purpose. In this section, we conduct parameter sensitivity analysis to investigate how the generalization performance of our proposed instance-specific loss-weighted decoding strategy changes with K.

Fig. 2 illustrates the performance fluctuation of ILW-Exp for both binary ECOC and ternary ECOC. It is shown that the performance of ILW-Exp is relatively stable when the value of K varies from 8 to 12. In this paper, we fix $K = 10$ in all cases which can also be used as the default parameter.

V. Application to Softmax Regression

Softmax regression is a generalized version of logistic regression for multi-class classification where logistic regression was initially designed for binary classification. Generally, it is regarded as one algorithm in direct strategy which can directly learn from multi-class data. In this section, we show that softmax regression can be considered as working based on OvR decomposition, but the decomposed binary classification problems are solved in a joint manner. Thus, the final multi-class prediction can be determined with the help of our proposed instance-specific loss-weighted decoding strategy instead of simply maximizing the predicted confidence.

A. Methodology

Softmax regression aims to estimate each class's probability for each sample $\boldsymbol{x}_i$ by learning a set of model parameters $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N] \in \mathbb{R}^{d\times N}$ as follows:

$$\begin{bmatrix} P(y_i = c_1 \mid \boldsymbol{x}_i) \\ P(y_i = c_2 \mid \boldsymbol{x}_i) \\ \vdots \\ P(y_i = c_N \mid \boldsymbol{x}_i) \end{bmatrix} = \frac{1}{\sum_{j=1}^{N} e^{\boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}_i}} \begin{bmatrix} e^{\boldsymbol{\theta}_1^{\mathrm{T}} \boldsymbol{x}_i} \\ e^{\boldsymbol{\theta}_2^{\mathrm{T}} \boldsymbol{x}_i} \\ \vdots \\ e^{\boldsymbol{\theta}_N^{\mathrm{T}} \boldsymbol{x}_i} \end{bmatrix} \tag{24}$$

Note that each class's probability is related to all parameters in $\boldsymbol{\Theta}$ due to the normalization term $\sum_{j=1}^{N} e^{\boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}_i}$. However, if we rank these N probabilities, the ranking of the j-th probability $P(y_i = c_j \mid \boldsymbol{x}_i)$ is only dependent on $\boldsymbol{\theta}_j$ $(1 \le j \le N)$ as the normalization term functions equally for all probabilities.

Generally, $\boldsymbol{\Theta}$ is determined by maximum likelihood estimation (MLE) where the likelihood function is defined as follows:

$$L(\boldsymbol{\Theta}) = \prod_{i=1}^{m} \prod_{j=1}^{N} [P(y_i = c_j \mid \boldsymbol{x}_i)]^{[\![y_i = c_j]\!]} \tag{25}$$

To prevent numerical underflow caused by continuous multiplication in Eq.(25), a common practice is to optimize the log-likelihood $LL(\boldsymbol{\Theta}) = \ln L(\boldsymbol{\Theta})$. Moreover, maximizing $LL(\boldsymbol{\Theta})$ is equivalent to minimizing $-LL(\boldsymbol{\Theta})$. Then, the optimization problem for softmax regression corresponds to:

$$\min_{\boldsymbol{\Theta}} - \sum_{i=1}^{m} \sum_{j=1}^{N} [\![y_i = c_j]\!] \cdot \ln P(y_i = c_j \mid \boldsymbol{x}_i) \tag{26}$$

It can be observed that the above optimization problem determines all $\boldsymbol{\theta}_j$s in a joint manner. The obtained solution must make the sum of N probabilities be equal to one.

After model parameters are obtained via optimizing Eq.(26), the final multi-class prediction for unseen instance $\boldsymbol{x}_*$ can be determined as follows:

$$y_* = c_{\hat{j}}, \text{ where } \hat{j} = \operatorname*{arg\,max}_{1 \le j \le N} \; P(c_j \mid \boldsymbol{x}_*) \tag{27}$$

From the above derivation, softmax regression essentially operates in an OvR decomposition mode. During the encoding stage, it decomposes the original multi-class problem into N binary classification problems, aiming to learn a linear model $\boldsymbol{\theta}_j$ for each class $(1 \le j \le N)$. In the decoding stage, the determination formula in Eq.(27) is identical to the maximum confidence decoding in Eq.(13) for the OvR strategy, provided we set $f_j^r(\boldsymbol{x}_*) = P(c_j \mid \boldsymbol{x}_*)$. The most significant difference lies in the training stage, specifically how the model parameters $\boldsymbol{\theta}_j$ for each class are determined. Generally, OvR independently learns the model parameters for each class. In contrast, as shown in Eq.(26), softmax regression determines all $\boldsymbol{\theta}_j$s jointly. Thus, softmax regression can indeed be considered as an OvR decomposition-based multi-class classifier with jointly learning all model parameters. As a result, further improvements in its performance may be achieved through our instance-specific loss-weighted decoding.

Specifically, it is easy to know that Eq.(27) is equivalent to the following formulation:

$$y_* = c_{\hat{j}}, \text{ where } \hat{j} = \operatorname*{arg\,max}_{1 \le j \le N} \; \boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}_* \tag{28}$$

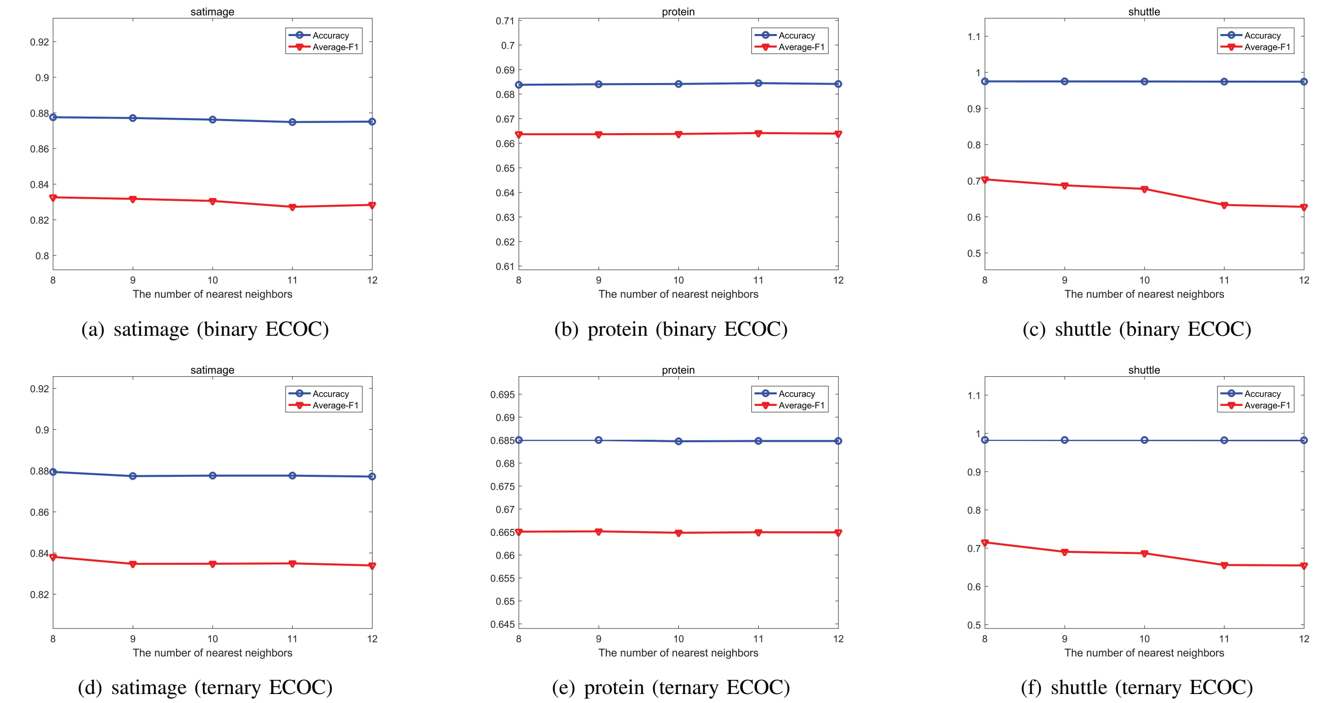
As $P(c_j \mid \boldsymbol{x}_*) \in [0, 1]$ and $\boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}_* \in \mathbb{R}$, thus we set $f_j^r(\boldsymbol{x}_*)$ to the latter real-valued inner product instead of the former normalized probability, i.e., $f_j^r(\boldsymbol{x}_*) = \boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}_*$.

To estimate the instance-specific empirical accuracy of the l-th binary classifier f_l $(1 \le l \le N)$ for unseen instance $\boldsymbol{x}_*$, the binary prediction for training samples in its K nearest neighbors should be determined. Let $(\boldsymbol{x}_{i_k}^l, y_{i_k}^l)$ be the k-th nearest neighbor $(1 \le k \le K)$, we determine its binary prediction as $f_l^b(\boldsymbol{x}_{i_k}^l) = \text{sign}(f_l^r(\boldsymbol{x}_{i_k}^l))$ with the sign function.

With the above adaptions, we can apply instance-specific loss-weighted decoding in Section IV for softmax regression.

B. Experiments

1) Experimental Setup: Following the experimental setup in Section IV-B1, we also employ the fifteen datasets in Table II to construct the testbed and the two evaluation metrics defined in Eq.(22) and Eq.(23) to measure one multi-class classifier's performance.



(a) satimage (binary ECOC)
(b) protein (binary ECOC)
(c) shuttle (binary ECOC)
(d) satimage (ternary ECOC)
(e) protein (ternary ECOC)
(f) shuttle (ternary ECOC)

Fig. 2. Parameter sensitivity analysis w.r.t. the number of nearest neighbors for instance-specific loss-weighted decoding.

TABLE XI
EXPERIMENTAL RESULTS (MEAN±STD.) FOR SOFTMAX REGRESSION WHERE THE BEST PERFORMANCE FOR EACH DATASET IS HIGHLIGHTED IN BOLDFACE.

	Accuracy			*Average*-F1		
Dataset	Max. Conf.	ILW-0/1	ILW-Exp	Max. Conf.	ILW-0/1	ILW-Exp
iris	0.933±0.070	**0.940±0.066**	**0.940±0.066**	0.930±0.070	**0.939±0.065**	**0.939±0.065**
wine	**0.983±0.027**	0.949±0.049	0.977±0.039	**0.977±0.037**	0.943±0.055	0.972±0.048
glass	0.603±0.108	0.648±0.132	**0.649±0.094**	0.528±0.123	0.565±0.170	**0.576±0.143**
svmguide2	0.826±0.049	0.762±0.080	**0.829±0.055**	**0.745±0.104**	0.664±0.138	0.743±0.111
vowel	0.635±0.061	0.585±0.053	**0.708±0.045**	0.599±0.061	0.557±0.053	**0.674±0.035**
dna	**0.937±0.016**	0.916±0.016	**0.937±0.016**	**0.928±0.019**	0.899±0.021	**0.928±0.019**
segment	0.932±0.020	0.927±0.024	**0.948±0.015**	0.932±0.017	0.925±0.023	**0.947±0.012**
satimage	0.860±0.021	0.869±0.018	**0.872±0.020**	0.812±0.025	0.821±0.024	**0.828±0.026**
usps	0.953±0.008	0.938±0.009	**0.962±0.008**	0.947±0.009	0.932±0.010	**0.958±0.010**
pendigits	0.959±0.008	0.977±0.004	**0.978±0.004**	0.959±0.007	**0.977±0.004**	**0.977±0.004**
letter	0.767±0.014	0.813±0.016	**0.864±0.015**	0.762±0.014	0.809±0.015	**0.861±0.015**
protein	**0.684±0.013**	0.625±0.014	0.682±0.013	**0.665±0.014**	0.612±0.014	**0.665±0.014**
poker	0.500±0.010	0.501±0.007	**0.524±0.007**	0.082±0.008	0.128±0.014	**0.129±0.013**
shuttle	0.966±0.002	**0.998±0.001**	0.997±0.001	0.602±0.068	**0.817±0.100**	0.770±0.097
mnist	0.915±0.003	0.907±0.003	**0.933±0.002**	0.914±0.003	0.905±0.003	**0.932±0.002**

TABLE XII
WILCOXON SIGNED-RANKS TEST FOR SOFTMAX REGRESSION (SIGNIFICANCE LEVEL $\alpha = 0.1$; p-VALUES SHOWN IN THE BRACKETS)

	Accuracy	*Average*-F1
ILW-0/1 vs.		
Max. Conf.	**tie**[5.24e-01]	**tie**[9.78e-01]
ILW-Exp vs.		
Max. Conf.	**win**[1.16e-03]	**win**[2.62e-03]
ILW-0/1	**win**[6.10e-04]	**win**[5.25e-03]

We aim to compare the vanilla softmax regression that determines the final prediction via Eq.(27) and the improved softmax regression that determines the final prediction via instance-specific loss-weighted decoding. Zero-one loss and exponential loss are also investigated as loss function. For convenience, the vanilla softmax regression is denoted as "Max. Conf.". Following the notations in previous section, the two improved softmax regression versions are also denoted as ILW-0/1 and ILW-Exp, respectively.

2) Experimental Results: Table XI reports the detailed experimental results. To facilitate comparison, we also highlight the best performance for each dataset in boldface. Moreover, to compare the overall performance over benchmark datasets, Wilcoxon signed-ranks test [8] is employed as the statistical tool. Table XII summarizes the corresponding test results, where the win/tie/loss in each cell for "A vs. B" means that the decoding strategy A achieves superior/comparable/inferior performance against the decoding strategy B.

According to the reported experimental results, we can make the following observations:

- According to Theorem 2 in Section III-C, "Max. Conf." is equivalent to LB-Exp because exponential loss is monotonically decreasing. As shown in Table XII, ILW-Exp achieves superior performance against "Max. Conf." in terms of both evaluation metrics. These experimental results further validate the effectiveness of instance-specific loss-weighted decoding.
- As discussed in Section III-C, Hamming decoding (i.e., LB-0/1) is not suitable for OvR decomposition. With the help of our decoding strategy, the improved version ILW-0/1 of Hamming decoding can achieve comparable performance against "Max. Conf." in terms of both evaluation metrics. These experimental results can also be regarded as another evidence for the effectiveness of instance-specific loss-weighted decoding.
- It is also worth noting that ILW-Exp achieves superior performance against ILW-0/1. As discussed in Section IV-B2, loss function generally has no significant influence on the final performance for different ECOC versions. But for softmax regression in this section, zero-one loss is not recommended.

VI. Conclusion

In this paper, we propose to consider the instance-specific generalization performance of each decomposed binary classifier for loss-weighted decoding, leading to the instance-specific loss-weighted decoding strategy. Experiments validate the effectiveness of the proposed decoding strategy, when loss function is exponential loss that is the most commonly-used in loss-based decoding as well as the zero-one loss which is also commonly used (i.e., Hamming decoding) but is rarely mentioned explicitly. We also propose to apply the proposed instance-specific loss-weighted decoding for softmax regression which can be regarded as working based on OvR decomposition. Experiments show that the performance of softmax regression can indeed be further improved with our proposed decoding strategy.

There are at least two issues that can be further explored. From the algorithm level, one limitation of the proposed decoding strategy is that the instance-specific generalization performance is still estimated with binary-valued prediction for the proposed decoding strategy. In the future, real-valued predicted confidences can be exploited for the estimation. From the application level, we are in the era of deep learning, since the output layer in neural networks for multi-class classification usually takes the same working mechanism with softmax regression, it is very interesting to investigate whether the proposed decoding strategy can still improve the performance of neural network-based models.

References

[1] S. Aeeneh, N. Zlatanov, and J. Yu, "New bounds on the accuracy of majority voting for multiclass classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[2] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *Journal of Machine Learning Research*, vol. 1, pp. 113–141, 2000.

[3] D. Bohning, "Multinomial logistic regression algorithm," *Annals of the Institute of Statistical Mathematics*, vol. 44, no. 1, pp. 197–200, 1992.

[4] Y. Chen, Z. Wen, B. He, and J. Chen, "Efficient decomposition selection for multi-class classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3751–3764, 2023.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[6] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," *Machine Learning*, vol. 47, pp. 201–233, 2002.

[7] P. del Moral, S. Nowaczyk, and S. Pashami, "Why is multiclass classification hard?" *IEEE Access*, vol. 10, pp. 80 448–80 462, 2022.

[8] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[9] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1995.

[10] K. Duan and S. S. Keerthi, "Which is the best multiclass SVM method? an empirical study," in *Proceedings of the 6th International Workshop on Multiple Classifier Systems*. Seaside, CA, USA: Springer, 2005, pp. 278–285.

[11] S. Escalera, O. Pujol, and P. Radeva, "On the decoding process in ternary error-correcting output codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 120–134, 2010.

[12] S. Escalera, D. M. J. Tax, O. Pujol, P. Radeva, and R. P. W. Duin, "Subclass problem-dependent design for error-correcting output codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 6, pp. 1041–1054, 2008.

[13] I. Evron, O. Onn, T. W. Orzech, H. Azeroual, and D. Soudry, "The role of codeword-to-class assignments in error-correcting codes: An empirical study," in *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*. Palau de Congressos, Valencia, Spain: PMLR, 2023, pp. 8053–8077.

[14] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[15] K. Feng, S. Liong, and K. Liu, "The design of variable-length coding matrix for improving error correcting output codes," *Information Science*, vol. 534, pp. 192–217, 2020.

[16] V. Feofanov, E. Devijver, and M.-R. Amini, "Multi-class probabilistic bounds for majority vote classifiers with partially labeled data," *Journal of Machine Learning Research*, vol. 25, pp. 1–47, 2024.

[17] C. Ferng and H. Lin, "Multilabel classification using error-correcting codes of hard or soft bits," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 11, pp. 1888–1900, 2013.

[18] A. Frid, L. M. Manevitz, and O. Mosafi, "Multi-class classification in parkinson's disease by leveraging internal topological structure of the data and of the label space," in *Proceedings of the International Joint Conference on Neural Networks*, Budapest, Hungary, 2019, pp. 1–9.

[19] J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, "Multilabel classification via calibrated label ranking," *Machine Learning*, vol. 73, no. 2, pp. 133–153, 2008.

[20] S. Gupta and S. Amin, "Scalable design of error-correcting output codes using discrete optimization with graph coloring," in *Advances in Neural Information Processing Systems 35*, New Orleans, LA, USA, 2022.

[21] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[22] B.-B. Jia, J.-Y. Liu, J.-Y. Hang, and M.-L. Zhang, "Learning label-specific features for decomposition-based multi-class classification," *Frontiers of Computer Science*, vol. 17, no. 6, Art. no. 176348, 2023.

[23] B.-B. Jia and M.-L. Zhang, "Multi-dimensional classification via stacked dependency exploitation," *Science China Information Sciences*, vol. 63, no. 12, Art. no. 222102, 2020.

[24] B.-B. Jia and M.-L. Zhang, "Multi-dimensional classification via selective feature augmentation," *Machine Intelligence Research*, vol. 19, no. 1, pp. 38–51, 2022.

[25] B.-B. Jia and M.-L. Zhang, "Multi-dimensional classification: paradigm, algorithms and beyond," *Vicinagearth*, vol. 1, Art. no. 3, 2024.

[26] L. Jin, L. Zhang, and L. Zhao, "Max-difference maximization criterion: a feature selection method for text categorization," *Frontiers of Computer Science*, vol. 17, no. 1, Art. no. 171337, 2023.

[27] T. Kajdanowicz and P. Kazienko, “Multi-label classification using error correcting output codes,” *International Journal of Applied Mathematics and Computer Science*, vol. 22, no. 4, pp. 829–840, 2012.

[28] S. Kang, S. Cho, and P. Kang, “Constructing a multi-class classifier using one-against-one approach with different binary classifiers,” *Neurocomputing*, vol. 149, pp. 677–682, 2015.

[29] D. G. Kleinbaum and M. Klein, *Logistic regression: A self-learning text*. New York, NY, USA: Springer, 2010.

[30] P. Li and H. Liu, “Binary decomposition for multi-class classification problems: Development and applications,” in *Proceedings of the 2023 International Conference on Machine Learning and Cybernetics*. Adelaide, Australia: IEEE, 2023, pp. 452–457.

[31] S. Li, L. Song, X. Wu, Y.-M. Cheung, and X. Yao, “Multi-class imbalance classification based on data distribution and adaptive weights,” *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[32] Y. Liang, C. Liu, H. Wang, K. Liu, J. Yao, Y. She, G. Dai, and Y. Okina, “A novel error-correcting output codes based on genetic programming and ternary digit operators,” *Pattern Recognition*, vol. 110, Art. no. 107642, 2021.

[33] G. Lin, J. Gao, N. Zeng, Y. Xu, K. Liu, B. Wang, J. Yao, and Q. Wu, “A self-adaptive soft-recoding strategy for performance improvement of error-correcting output codes,” *Pattern Recognition*, vol. 143, Art. no. 109813, 2023.

[34] G. Lin, K. Liu, B. Wang, and X. Zhang, “Partial label learning based on label distributions and error-correcting output codes,” *Soft Computing*, vol. 25, no. 2, pp. 1049–1064, 2021.

[35] G. Lin, Z. Xiao, J. Liu, B. Wang, K. Liu, and Q. Wu, “Feature space and label space selection based on error-correcting output codes for partial label learning,” *Information Sciences*, vol. 589, pp. 341–359, 2022.

[36] W. Lin, Q. Ge, S. Liong, J. Liu, K. Liu, and Q. Wu, “The design of error-correcting output codes based deep forest for the micro-expression recognition,” *Applied Intelligence*, vol. 53, no. 3, pp. 3488–3504, 2023.

[37] B.-Q. Liu, B.-B. Jia, and M.-L. Zhang, “Towards enabling binary decomposition for partial multi-label learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 13 203–13 217, 2023.

[38] J.-Y. Liu and B.-B. Jia, “Combining one-vs-one decomposition and instance-based learning for multi-class classification,” *IEEE Access*, vol. 8, pp. 197 499–197 507, 2020.

[39] K. Liu, J. Gao, Y. Xu, K. Feng, X. Ye, S. Liong, and L. Chen, “A novel soft-coded error-correcting output codes algorithm,” *Pattern Recognition*, vol. 134, Art. no. 109122, 2023.

[40] K. Liu, X. Ye, H. Guo, Q. Wu, and Q. Hong, “The design of soft recoding-based strategies for improving error-correcting output codes,” *Applied Intelligence*, vol. 52, no. 8, pp. 8856–8873, 2022.

[41] M. Liu, D. Zhang, S. Chen, and H. Xue, “Joint binary classifier learning for ECOC-based multi-class classification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2335–2341, 2016.

[42] W. Liu, H. Wang, X. Shen, and I. W. Tsang, “The emerging trends of multi-label learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7955–7974, 2022.

[43] R. Maclin, “Boosting classifiers regionally,” in *Proceedings of the 15th AAAI Conference on Artificial Intelligence*. Madison, WI, USA: AAAI Press, 1998, pp. 700–705.

[44] A. Moreo, A. Esuli, and F. Sebastiani, “Word-class embeddings for multiclass text classification,” *Data Mining and Knowledge Discovery*, vol. 35, no. 3, pp. 911–963, 2021.

[45] H. D. Nguyen, L. J. Lavalva, S. Ho, M. S. Khan, and N. Kaegi, “Optimal n-ary ECOC matrices for ensemble classification,” in *Proceedings of the IEEE Symposium Series on Computational Intelligence*. Orlando, FL, USA: IEEE, 2021, pp. 1–8.

[46] Z. Ning, Z. Jiang, and D. Zhang, “To combat multiclass imbalanced problems by aggregating evolutionary hierarchical classifiers,” *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[47] A. Passerini, M. Pontil, and P. Frasconi, “New results on error correcting output codes of kernel machines,” *IEEE Transactions on Neural Networks*, vol. 15, no. 1, pp. 45–54, 2004.

[48] P. Pawara, E. Okafor, M. Groefsema, S. He, L. R. B. Schomaker, and M. A. Wiering, “One-vs-one classification for deep neural networks,” *Pattern Recognition*, vol. 108, Art. no. 107528, 2020.

[49] O. Pujol, S. Escalera, and P. Radeva, “An incremental node embedding technique for error correcting output codes,” *Pattern Recognition*, vol. 41, no. 2, pp. 713–725, 2008.

[50] O. Pujol, P. Radeva, and J. Vitrià, “Discriminant ECOC: A heuristic method for application dependent design of error correcting output codes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 6, pp. 1007–1012, 2006.

[51] A. Rocha and S. K. Goldenstein, “Multiclass from binary: Expanding one-versus-all, one-versus-one and ECOC-based approaches,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 289–302, 2014.

[52] P. Rodríguez, M. Á. Bautista, J. Gonzàlez, and S. Escalera, “Beyond one-hot encoding: Lower dimensional target embedding,” *Image and Vision Computing*, vol. 75, pp. 21–31, 2018.

[53] J. Shan, C. Hou, H. Tao, W. Zhuge, and D. Yi, “Randomized multi-label subproblems concatenation via error correcting output codes,” *Neurocomputing*, vol. 410, pp. 317–327, 2020.

[54] Y.-X. Shi, D.-B. Wang, and M.-L. Zhang, “Partial label learning with gradually induced error-correction output codes,” in *Proceedings of the 29th International Conference on Neural Information Processing*. Virtual Event: Springer, 2022, pp. 200–211.

[55] Y. Shiraishi and K. Fukumizu, “Statistical approaches to combining binary classifiers for multi-class classification,” *Neurocomputing*, vol. 74, no. 5, pp. 680–688, 2011.

[56] Y. Song, Q. Kang, and W. P. Tay, “Error-correcting output codes with ensemble diversity for robust learning in neural networks,” in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*. Virtual Event: AAAI Press, 2021, pp. 9722–9729.

[57] V. Subramanian, R. Arya, and A. Sahai, “Generalization for multiclass classification with overparameterized linear models,” in *Advances in Neural Information Processing Systems 35*, New Orleans, LA, USA, 2022.

[58] G. Szücs, “Multiclass classification by min-max ECOC with hamming distance optimization,” *The Visual Computer*, vol. 39, no. 9, pp. 3949–3961, 2023.

[59] L. Wang, H. Wei, Y. Zheng, J. Dong, and G. Zhong, “Deep error-correcting output codes,” *Algorithms*, vol. 16, no. 12, Art. no. 555, 2023.

[60] Z. Wang and X. Xue, “Multi-class support vector machine,” in *Support Vector Machines Applications*, Y. Ma and G. Guo, Eds. Cham: Springer, 2014, pp. 23–48.

[61] K. Wei, T. Li, F. Huang, J. Chen, and Z. He, “Cancer classification with data augmentation based on generative adversarial networks,” *Frontiers of Computer Science*, vol. 16, no. 2, Art. no. 162601, 2022.

[62] X.-S. Wei, S.-L. Xu, H. Chen, L. Xiao, and Y. Peng, “Prototype-based classifier learning for long-tailed visual recognition,” *Science China Information Sciences*, vol. 65, no. 6, pp. 1–15, 2022.

[63] T. Windeatt and R. Ghaderi, “Coding and decoding strategies for multi-class learning problems,” *Information Fusion*, vol. 4, no. 1, pp. 11–21, 2003.

[64] K. Wu, F. Jia, and Y. Han, “Domain-specific feature elimination: multi-source domain adaptation for image classification,” *Frontiers of Computer Science*, vol. 17, no. 4, Art. no. 174705, 2023.

[65] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. F. M. Ng, B. Liu, P. S. Yu, Z. Zhou, M. S. Steinbach, D. J. Hand, and D. Steinberg, “Top 10 algorithms in data mining,” *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[66] S. Xie, Z. He, L. Pan, K. Liu, and S. Su, “An adaptive error-correcting output codes algorithm based on gene expression programming and similarity measurement matrix,” *Pattern Recognition*, vol. 145, Art. no. 109957, 2024.

[67] Y. Yang, Z. Sun, H. Zhu, Y. Fu, Y. Zhou, H. Xiong, and J. Yang, “Learning adaptive embedding considering incremental class,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 2736–2749, 2023.

[68] X. Ye, K. Liu, and S. Liong, “A ternary bitwise calculator based genetic algorithm for improving error correcting output codes,” *Information Science*, vol. 537, pp. 485–510, 2020.

[69] B. Zhang, J. Zhu, and H. Su, “Toward the third generation artificial intelligence,” *Science China Information Sciences*, vol. 66, no. 2, Art. no. 121101, 2023.

[70] H. Zhang, J. T. Zhou, T. Wang, I. W. Tsang, and R. S. M. Goh, “Deep n-ary error correcting output codes,” in *Proceedings of the 13th EAI International Conference on Mobile Multimedia Communications*. Harbin, China: EAI, 2020, pp. 409–427.

[71] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, and X. Geng, “Binary relevance for multi-label learning: An overview,” *Frontiers of Computer Science*, vol. 12, no. 2, pp. 191–202, 2018.

[72] M.-L. Zhang, F. Yu, and C.-Z. Tang, “Disambiguation-free partial label learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 10, pp. 2155–2167, 2017.

[73] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[74] F. Zheng and H. Xue, "Subclass maximum margin tree error correcting output codes," in *Proceedings of the 15th Pacific Rim International Conference on Artificial Intelligence*, Nanjing, China, 2018, pp. 454–462.

[75] F. Zheng, H. Xue, X. Chen, and Y. Wang, "Maximum margin tree error correcting output codes," in *Proceedings of the 14th Pacific Rim International Conference on Artificial Intelligence*, Phuket, Thailand, 2016, pp. 681–691.

[76] D. Zhou, Y. Yang, and D. Zhan, "Learning to classify with incremental new class," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 6, pp. 2429–2443, 2022.

[77] J. T. Zhou, I. W. Tsang, S. Ho, and K. Müller, "N-ary decomposition for multi-class classification," *Machine Learning*, vol. 108, no. 5, pp. 809–830, 2019.

[78] Z.-H. Zhou, *Machine mearning*. Singapore: Springer Nature, 2021.