2025 International Conference on Machine Learning and Cybernetics (ICMLC) | 979-8-3315-8736-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICMLC66258.2025.11280112

HEURISTIC SETTINGS OF VIRTUAL ORDERS OF CLASS LABELS FOR NOMINAL CLASSIFICATION

HAN LIU[1], YINGHUI PAN[2], HAO CHEN[3]*

[1]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China
[2]School of Artificial Intelligence, Shenzhen University, Shenzhen 518060, China
[3]Information Center, Shenzhen University, Shenzhen 518060, China
E-MAIL: han.liu@szu.edu.cn, panyinghui@szu.edu.cn, haochen@szu.edu.cn
*Corresponding Author

Abstract:

In multi-class classification tasks, binary decomposition of the class attribute is necessary for some learning algorithms such as support vector machines. While the class attribute is nominal, the binary decomposition is typically done in an unordered way, which can result in a negative impact on the effectiveness and efficiency of learning binary classifiers in some scenarios. In this paper, we explore whether it is achievable to improve the performance of nominal classification by setting virtual class orders for achieving ordered binary decomposition of the class attribute. Specifically, we propose a framework to search for virtual label orders that are suitable for effectively transforming a nominal classification problem into simpler binary ones, where the framework consists of a heuristic search module for searching local optimal orders and an ensemble module for combining the orders. The experimental results show that our framework leads to an improvement of the classification performance in comparison with arbitrarily assigning a label order or setting the class labels to be unordered. The current work can be regarded as a type of class representation learning.

Keywords:

Classification; Binary decomposition; Virtual class order

1. Introduction

In a multi-class classification task, binary decomposition of the class attribute can be adopted to transform the original classification problem into a series of simpler problems [24, 22]. This kind of decomposition is highly required for those algorithms that cannot directly learn a multi-class classifier [1, 16].

Existing studies indicate that binary decomposition can be adopted in an ordered or unordered way depending on whether the class attribute is ordinal or not [21]. In an ordinal classification task, an unordered way of binary decomposition can be adopted practically but it is considered less effective than an ordered one. Specifically, as stressed by [10], it is reasonable to adopt ordered binary decomposition instead of unordered one for achieving a performance improvement, if the order of the class labels is also present in the feature space. In this context, ordered decomposition results in simpler binary classification problems, since a label order naturally indicates that the distance between two low-ranked (or high-ranked) classes are smaller than the distance between a low-ranked class and a high-ranked one [18, 23], and thus it seems easier to discriminate lower-ranked classes from higher-ranked classes than to discriminate an arbitrary label subset from its complement. On the other hand, ordered decomposition generally results in lower computational complexity than unordered decomposition [15, 8]. Although there is not a true order of class labels in nominal classification, based on the concept of virtual orders [12], we explore whether it is achievable to improve the classification performance by setting a virtual label order to enable the adoption of ordered binary decomposition. In this context, the virtual label order is expected to reflect that the separability between two classes assigned similar ranks is worse than that between two classes assigned very different ranks, in order to promote an improvement of the classification performance.

The contributions of this paper include the following points:

- We formulate a constrained optimization problem in the context of label order search and propose a framework that involves the heuristic search module and the ensemble module for selecting and combining label orders.

979-8-3315-8736-9/25/$31.00 ©2025 IEEE

- The heuristic search module is designed by modifying the velocity update strategy of particle swarm optimization (PSO) to better suit our formulated problem on search of the label order.
- Based on the search results returned from the heuristic search module, we propose to create ensemble diversity by training classifiers using different label orders that show top-ranked fitness.
- The results indicate that the proposed framework leads to an improvement of the classification performance in comparison with selecting a label order arbitrarily or setting the class attribute to be nominal.

2 Problem Formulation

For a data set containing k classes, there are totally $k!$ possible virtual orders of the class labels. Each possible order is represented as a k-dimensional vector $l_o = [l_{o1}, l_{o2}, \ldots, l_{ok}]$, $l_{oi} = 1, 2, \ldots, k, i, j = 1, 2, \ldots, k, \forall i \neq j : l_{oi} \neq l_{oj}$, which is treated as a candidate solution in the context of heuristic search. Therefore, the problem $\mathcal{T}$ that this paper aims to solve is defined as an optimization problem, in a k-dimensional search space $\mathcal{S}$, i.e., the aim is to find the optimal solution among the $k!$ candidate solutions, using the fitness function $fitness_D$ evaluated by measuring classification performance on data set D, as illustrated in Eq. (1).

$$\begin{aligned} \max_{o=1,2,\ldots,k!} fitness_D(l_o) \qquad & s.t. \\ 1 \leq l_{oi} \leq k, i = 1, 2, \ldots, k; & \\ \forall i \neq j : l_{oi} \neq l_{oj}, i, j = 1, 2, \ldots, k. & \end{aligned} \tag{1}$$

Since the search space is made up of $k!$ discrete points, $\mathcal{T}$ is essentially a combinatorial optimization problem [7]. Due to the constraints that the values of various dimensions of the above-mentioned k-dimensional vector l_o need to be integers and are mutually different, any points (k-dimensional vectors) that do not satisfy the condition $\forall i \neq j : l_{oi} \neq l_{oj}$ must not be included in the search space $\mathcal{S}$. In such a situation, $\mathcal{T}$ is defined as a non-convex optimization problem. Specifically, given the definition that an optimization problem is convex if and only if the objective function is convex and the domain of the function is described as a convex set. In the context of our formulated problem shown in Eq. (1), the objective function is subject to a constraint on its domain $\mathcal{C}$ and it can be derived that $\mathcal{C}$ is not a convex set, as illustrated below: suppose that there are k classes ($k \geq 2$), so $\mathcal{C}$ contains $k!$ k-dimensional vectors and each vector must satisfy the conditions shown in Eq. (1). Given $\theta \in [0, 1]$, we randomly select two k-dimensional vectors from $\mathcal{C}$, say $l_1 = [1, 2, \ldots, k]$ and $l_2 = [k, k-1, \ldots, 1]$. The following operation shows $\theta l_1 + (1-\theta) l_2 \notin \mathcal{C}$:

$$\begin{aligned} \theta l_1 + (1-\theta) l_2 &= \theta[1, 2, \ldots, k] + (1-\theta)[k, k-1, \ldots, 1] \\ &= [\theta + k(1-\theta), (3-k)\theta + (k-1), \ldots, (k-1)\theta + 1] \end{aligned}$$

where the last dimension of the above-derived vector is ranged in $[1, k]$, i.e., $0 \leq \theta \leq 1 \implies 1 \leq (k-1)\theta + 1 \leq k$, which does not satisfy the condition that each dimension of the k-dimensional vector $l_o \in \mathcal{C}$ must be an integer between 1 and k. More generally, if $\mathcal{C}$ is assumed to be a convex set, then it is supposed to be satisfied that $\forall l_o = [l_{o1}, l_{o2}, \ldots, l_{ok}], \forall l_{o'} = [l_{o'1}, l_{o'2}, \ldots, l_{o'k}], \forall i = 1, 2, \ldots, k, \forall \theta \in [0, 1] : \theta l_{oi} + (1-\theta) l_{o'i} \in \{1, 2, \ldots, k\}$. However, we have:
$\theta = 0 \implies \theta l_{oi} + (1-\theta) l_{o'i} = l_{o'i}$; $\theta = 1 \implies \theta l_{oi} + (1-\theta) l_{o'i} = l_{oi}$; $0 < \theta < 1 \implies \theta l_{oi} + (1-\theta) l_{o'i}$ may not be an integer.

The above derivation shows a contradiction case: $\exists i = 1, 2, \ldots, k, \exists \theta \in [0, 1] : l_{oi} \neq l_{o'i}, \theta l_{oi} + (1-\theta) l_{o'i} \notin \{1, 2, \ldots, k\} \implies \theta l_o + (1-\theta) l_{o'} \notin \mathcal{C}$. Therefore, it is proved that $\mathcal{C}$ is not described as a convex set and thus $\mathcal{T}$ is defined as a non-convex constrained optimization problem.

3 Search and Combination of Label Orders

In this section, we present the procedure of our framework, which involves the heuristic search module and the ensemble module as illustrated in Fig. 1.

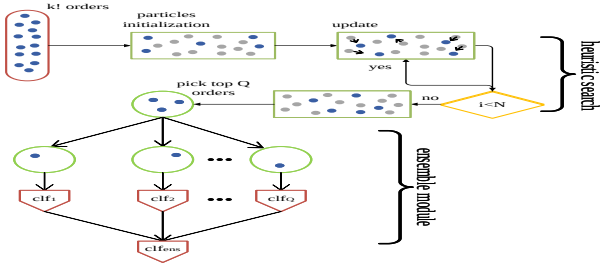

FIGURE 1. Overview of Proposed Framework

3.1 Heuristic Search Module

The heuristic search module shown in Algorithm 1 aims to identify a suitable label order, which involves modifying the velocity update strategy of the classic particle swarm optimization algorithm to better suit the search problem.

Algorithm 1: Heuristic Search Module

Input : Number of iterations N, Size of population M.
Output: Virtual order(s) of class labels.

For each particle a, initialize the velocity vector v, the position vector pv, the fitness value $fitness$, the best position vector of the individual $pbest$, and the best position vector of the population $gbest$.
for $t = 0$ *to* $N - 1$ **do**
  for $a = 0$ *to* $M - 1$ **do**
    update v_a, pv_a, $fitness_a$ and $pbest_a$ of particle a;
  end
  update the best position vector $gbest_t$ of the population;
end
if $gbest$ *shows significantly higher fitness than the second best one* **then**
  return $gbest$;
end
else
  return top Q traversed position vectors with similar fitness values;
end

The initial stage of the PSO algorithm [14] is to randomly initialize the positions of several particles in the population in a high-dimensional search space.

We pre-select $\frac{1}{2}k!$ label orders to form search space $\mathcal{S}$, i.e., any two of the candidate solutions in $\mathcal{S}$ do not happen to be two orders inverted to each other, such that the case of producing identical classifiers can be avoided (as analysed in Proposition 1 in Section 3.2). Furthermore, we take uniform sampling to obtain M initial solutions from $\mathcal{S}$ for promoting that these obtained solutions are sufficiently different. Each solution is represented as a k-dimensional order vector and is regarded as the position of a particle in the population. An initial velocity vector is randomly assigned to each particle. Corresponding to the order (position) vector, the dimensionality of the velocity vector is also equal to k, and the value of each dimension is an integer. The fitness of each particle is evaluated by using a validation set to measure the performance of the classification model trained using the label order (i.e., the position in which the particle is located). After the particle swarm is initialized, at each iteration t, the velocity vector $v_{a,j}(t)$ of each particle a is updated according to Eq. (2), through learning from the memory term $(\omega(t)v_{a,j}(t))$, the cognitive term $(c_1r_{1,j}(t)(y_{a,j}(t) - x_{a,j}(t)))$ and the environmental term $(c_2r_{2,j}(t)(\hat{y}_j(t) - x_{a,j}(t)))$, and then particle a moves from its current position $x_{a,j}(t)$ to another one according to Eq. (3).

$$v_{a,j}(t+1) = \omega(t)v_{a,j}(t) + c_1r_{1,j}(t)(y_{a,j}(t) - x_{a,j}(t)) + c_2r_{2,j}(t)(\hat{y}_j(t) - x_{a,j}(t)). \quad (2)$$

$$x_{a,j}(t+1) = x_{a,j}(t) + v_{a,j}(t+1). \quad (3)$$

In Eq. (2), $\omega(t)$ is the inertia parameter at iteration t. $v_{a,j}(t)$ is the value of the j-th element of the velocity vector of particle a at iteration t. c_1 and c_2 are acceleration constants. $r_{1,j}(t)$ and $r_{2,j}(t)$ are random numbers from 0 to 1 at iteration t. $x_{a,j}(t)$ represents the value of the j-th element of the position vector of particle a at iteration t. $y_{a,j}(t)$ represents the j-th element of the historical optimal position of the particle a before iteration $t + 1$. $\hat{y}_j(t)$ represents the j-th element of the global optimal position of the population before iteration $t + 1$.

Since the value $x_{a,j}(t)$ of each element of the position vector should be an integer between 1 and k, the corresponding value $v_{a,j}(t)$ needs to be an integer within the range of $[1 - x_{a,j}(t), k - x_{a,j}(t)]$, and the values of various elements of a velocity vector need to be partially different. However, according to Eq. (2), the velocity vector is not guaranteed to meet the above-mentioned requirement exactly, i.e., the values of some elements of the velocity vector may be decimals. For handling this situation, we propose to identify all possible velocity vectors based on the difference between the position of each particle a at iteration t and any possible positions of the particle at iteration $t + 1$, and select the velocity vector that is closest to $v_a(t + 1)$ for use at iteration $t + 1$.

After the completion of the heuristic search, statistical analysis is conducted to measure the significance of the fitness difference between the optimal order and each of the other orders. If the statistical analysis shows that the optimal order has a significant superiority in fitness in comparison with the other orders, the optimal order is adopted to set the order of the class labels of the data set. Otherwise, top Q label orders whose fitness values are not significantly different are combined through adopting the ensemble module presented in Section 3.2.

3.2 Ensemble Module

The ensemble module works by training classifiers using various label orders returned from the heuristic search module. In practice, ensemble pruning may be necessary due to the case that base classifiers trained using different orders may have some positive correlations. From this viewpoint, we analyse

theoretically in what situations and how the joint use of different label orders for training multiple classifiers may encourage or limit the ensemble diversity, based on Tumer and Ghosh's framework [20, 19]. Specifically, in the case of combining the probabilities predicted by different classifiers for each class, the expected added error of an ensemble is defined in Eq. (4), as analyzed by [19].

$$E_{add}^{ensemble} = E_{add}(\frac{1+\delta(T-1)}{T}), \tag{4}$$

where T is the number of classifiers, E_{add} is the expected added error of each single classifier and δ is a statistical measure of the correlation between class probability estimation errors of different classifiers [2, 11].

According to Eq. (4), it can be derived that the reduction of the correlation coefficient δ between estimation errors of different classifiers is an essential way of enhancing the diversity among these classifiers. Moreover, it can be derived according to [19] that the reduction of the correlation between errors of individual classifiers can be achieved by reducing the correlation between the probabilities estimated by these classifiers for each class. Our proposed strategy of correlation reduction is to train multiple classifiers using different virtual label orders alongside OrderedPartition [5]. Specifically, given a label set $\mathcal{L} = \{L_1, L_2, \ldots, L_k\}$ alongside the given virtual label order $L_1 \prec L_2 \prec \ldots \prec L_k$, $\mathcal{L}$ is decomposed into two subsets $\mathcal{L}_{Pos}^c$ and $\mathcal{L}_{Neg}^c$ in $k-1$ ways following the OrderedPartition strategy.

In the above setting, there are $k-1$ binary classifiers trained independently for predicting the class probabilities $P_e(Target > L_1|x_i)$, $P_e(Target > L_2|x_i)$, ..., $P_e(Target > L_{k-1}|x_i)$, respectively, given a new instance x_i. The probabilities of the k classes in $\mathcal{L}$ can be derived in the following way [5]:

$$\begin{aligned}
&P_e(L_1|x_i) = 1 - P_e(Target > L_1|x_i),\\
&P_e(L_2|x_i) = P_e(Target > L_1|x_i) \times (1 - P_e(Target > L_2|x_i)),\\
&\vdots\\
&P_e(L_k|x_i) = P_e(Target > L_{k-1}|x_i).
\end{aligned} \tag{5}$$

Since the probabilities $P_e(Target > L_1|x_i)$, $P_e(Target > L_2|x_i)$, ..., $P_e(Target > L_{k-1}|x_i)$ are predicted by $k-1$ binary classifiers trained independently using $k-1$ different distributions of the positive and negative classes, the $k-1$ estimated probabilities are considered to be independent and thus the derived probabilities of the k classes $L_1, L_2, \ldots, L_k$ are also considered to be independent. While an ensemble is created by training T classifiers (i.e., $T \times (k-1)$ binary subclassifiers) using T label orders, the T classifiers are diverse in some scenarios due to the diversification of the label order information. In particular, let us reformulate the derivation of the probability of each class $L_{\bar{c}}$ in Eq. (6).

$$P_e^q(L_{\bar{c}}|x_i) = \left\{ \begin{array}{l} 1 - P_e^q(Target > Rank_q(L_{\bar{c}})|x_i),\\ Rank_q(L_{\bar{c}}) = 1;\\ P_e^q(Target > Rank_q(L_{\bar{c}}) - 1|x_i) \times\\ (1 - P_e^q(Target > Rank_q(L_{\bar{c}})|x_i)),\\ 1 < Rank_q(L_{\bar{c}}) < k;\\ P_e^q(Target > Rank_q(L_{\bar{c}}) - 1|x_i),\\ Rank_q(L_{\bar{c}}) = k. \end{array} \right\} \tag{6}$$

where $Rank_q(L_{\bar{c}})$ represents the rank of class $L_{\bar{c}}$ in the q-th label order and $P_e^q(L_{\bar{c}}|x_i)$ denotes the posterior probability of class $L_{\bar{c}}$ predicted by using one or two binary sub-classifiers trained using the q-th label order.

According to Eq. (6), given the q-th and q'-th label orders, if one order is the inverted one of the other order, then we can have the following proposition:

Proposition 1 *While a fixed algorithm is used for learning, if the q-th label order is the inverted one of the q'-th order, then the probabilities $P_e^q(L_{\bar{c}}|x_i)$ and $P_e^{q'}(L_{\bar{c}}|x_i)$ that are estimated using the q-th and q'-th label orders are perfectly correlated for each class $L_{\bar{c}}, \bar{c} = 1, 2, \ldots, k$.*

Proof. For each class $L_{\bar{c}}$, if $Rank_q(L_{\bar{c}}) + Rank_{q'}(L_{\bar{c}}) = 1 + k$, we have:
$\exists q, q' : q \neq q', \forall \bar{c} = 1, 2, \ldots, k : Rank_q(L_{\bar{c}}) + Rank_{q'}(L_{\bar{c}}) = 1 + k \implies \forall R = 1, 2, \ldots, k-1 : \mathcal{L}_{Pos}^R(q) = \mathcal{L}_{Neg}^{1+k-R}(q'), \mathcal{L}_{Neg}^R(q) = \mathcal{L}_{Pos}^{1+k-R}(q'), \mathcal{D}_q(R) = \mathcal{D}_{q'}(1 + k - R) \implies \forall \bar{c} = 1, 2, \ldots, k : Corr(P_e^q(L_{\bar{c}}|x_i), P_e^{q'}(L_{\bar{c}}|x_i)) = 1$.

$\mathcal{L}_{Neg}^R(q)$ denotes the label subset which forms the negative class and is obtained by collecting the last R classes in the q-th label order. $\mathcal{D}_q(R)$ denotes the class distribution, which is obtained by merging the last R classes (or the rest) in the q-th label order as the negative (or positive) class. The above derivation indicates that the estimated probabilities $P_e^q(Target > Rank_q(L_{\bar{c}}) - 1|x_i)$ and $P_e^{q'}(Target > Rank_{q'}(L_{\bar{c}})|x_i)$ are negatively correlated for each class $L_{\bar{c}}$, i.e., $\forall \bar{c} = 1, 2, \ldots, k : P_e^q(Target > Rank_q(L_{\bar{c}}) - 1|x_i) + P_e^{q'}(Target > Rank_{q'}(L_{\bar{c}})|x_i) = 1$, so we reach the conclusion shown in Proposition 1 that the probabilities $P_e^q(L_{\bar{c}}|x_i)$ and $P_e^{q'}(L_{\bar{c}}|x_i)$ are perfectly correlated for each class $L_{\bar{c}}$.

Overall, in order to effectively enhance the diversity among individual classifiers, it is necessary to make the selected label orders hold the assumption: $\forall q \neq q' : \exists \bar{c} = 1, 2, \ldots, k : Rank_q(L_{\bar{c}}) + Rank_{q'}(L_{\bar{c}}) \neq 1 + k$, and increasing the number of classes on which the above assumption holds can generally

help better encourage the diversity among those classifiers in an ensemble.

4 Experimental Setup and Results Discussion

The details of the selected data sets are shown in Table 1. The experiment is conducted by 10 runs of 5-fold cross-validation on each data set and the average area under curve (AUC) and the standard deviation are taken for comparisons of various methods, namely, C4.5, One-vs-One (OVO) [9], One-vs-Rest (OVR) [17], Many-vs-Many (MVM) [4], random label ordering and the proposed framework.

The experimental comparisons aim to show that a transformation of a multi-class classification problems into a series of binary ones is meaningful, settings of virtual label orders help improve the effectiveness of the problem transformation and heuristic settings are more effective than random ones. In all cases, C4.5 is adopted for training binary or multi-class classifiers, where the default settings (provided in Weka [6]) of those hyper-parameters are used. The settings of all the hyper-parameters relevant to the proposed framework are shown in Table 2.

In the setting of the ensemble module, the Q orders returned from the heuristic search module are used for training base classifiers and ensemble pruning based on forward search [13, 3] is taken to select a sub-ensemble that leads to the best classification performance on validation data. For OVR, OVO and MVM, their implementations in Weka are adopted, where random coding is adopted as the strategy of error-correcting output codes (ECOC) [4] for MVM.

TABLE 1. Characteristics of data sets

Data sets	No. of features	No. of samples	No. of classes
Analcatdata	4	797	6
Car	6	1728	4
CMC	9	1473	3
Ecoli	7	327	5
Eucalyptus	19	736	5
First-order	51	6118	6
Glass	9	214	6
Jungle_chess	6	44819	3
Page-blocks	10	5473	5
Segment	19	2310	7

The results on the comparison of our proposed framework with the other methods are shown in Table 3, where the header "Random" of the third column represents the setting of an arbitrary virtual order of class labels. The results indicate that the proposed approach outperforms the other ones in most of the cases and setting virtual label orders is useful in improving the performance of nominal classification and the effectiveness of binary decomposition of the class attribute. However, the proposed approach may not be so effective while various class pairs in a data set show very similar inter-class separability, i.e., virtual label orders are not considered as useful information in the above situation.

TABLE 2. Hyperparameter settings

Dimensionality of Search Space: No. of classes
No. of particles in the swam, No. of iterations: 15 particles, 16 iterations, where No. of classes = 7 10 particles, 18 iterations, where No. of classes = 6 10 particles, 3 iterations, where No. of classes = 5 3 particles, 4 iterations, where No. of classes = 4 3 particles, 1 iterations, where No. of classes = 3
classifier: OrderedPartition + C4.5 (default hyper-parameters in Weka)
ω: max = 1, min = 0.8 (tuning by linearly decreasing)
c_1: max = 1.2, min = 0.1 (tuning by linearly decreasing)
c_2: max = 1.2, min = 0.1 (tuning by linearly increasing)
Base classifier: OrderedPartition + C4.5 (default hyper-parameters in Weka)
Ensemble pruning strategy: forward search
Fusion strategy: average of probabilities for each class

TABLE 3. Average AUC with standard deviation

Data sets	C4.5	Random	OVO	OVR	MVM	Proposed
Analcatdata	0.538±0.007	0.523±0.007	**0.572±0.006**	0.500±0.001	0.557±0.017	0.560±0.011
Car	0.990±0.003	0.990±0.004	0.987±0.002	0.994±0.003	0.992±0.002	**0.995±0.001**
CMC	0.664±0.009	0.669±0.010	0.655±0.008	0.684±0.006	**0.685±0.006**	0.684±0.009
Ecoli	0.918±0.007	0.915±0.012	0.942±0.005	0.919±0.011	**0.948±0.008**	**0.948±0.008**
Eucalyptus	0.799±0.012	0.768±0.014	**0.853±0.005**	0.769±0.008	0.831±0.017	0.841±0.020
First-order	0.738±0.005	0.746±0.007	0.791±0.002	0.763±0.003	0.793±0.008	**0.807±0.005**
Glass	0.809±0.018	0.809±0.024	0.864±0.009	0.821±0.025	0.863±0.014	**0.871±0.015**
Jungle_chess	0.955±0.001	0.954±0.003	0.899±0.001	**0.959±0.001**	0.956±0.005	0.958±0.001
Page-blocks	0.940±0.006	0.949±0.010	0.925±0.004	0.954±0.004	0.966±0.005	**0.971±0.005**
Segment	0.986±0.002	0.981±0.004	**0.991±0.001**	0.983±0.003	0.991±0.006	0.991±0.005

5. Conclusions

In this paper, we have formulated a new problem by defining the search of a suitable order as a task of combinatory optimization for improving the performance. Based on the problem formulation, we have proposed a framework, which involves a heuristic search module and an ensemble module. The experimental results show that the proposed framework results in an improvement of the classification performance in comparison with assigning a label order arbitrarily or setting the class attribute to be nominal. In the future, we will study how the extent to which the inter-class separabilities of various pairs of classes differ impacts the effectiveness of virtual order-driven binary decomposition.

Acknowledgements

This paper is supported by National Natural Science Foundation of China (Grants 62106147 and 62276168), Guangdong Provincial Natural Science Foundation (Grant 2023A1515010869)

References

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

[2] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[3] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 18, New York, NY, USA, 2004. Association for Computing Machinery.

[4] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2(1):263–286, 1994.

[5] E. Frank and M. Hall. A simple approach to ordinal classification. In *European Conference on Machine Learning*, pages 145–156. Springer, 2001.

[6] E. Frank, M. Hall, and I. H. Witten. Weka: The workbench for machine learning, 2016.

[7] H. Goto, K. Endo, M. Suzuki, Y. Sakai, T. Kanao, Y. Hamakawa, R. Hidaka, M. Yamasaki, and K. Tatsumura. High-performance combinatorial optimization based on classical mechanics. *Science Advances*, 7(6):eabe7953, 2021.

[8] P. A. Gutiérrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernandez-Navarro, and C. Hervas-Martinez. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):127–146, 2016.

[9] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *Advances in Neural Information Processing Systems 10, [NIPS Conference, Denver, Colorado, USA, 1997]*, 1997.

[10] J. C. Huhn and E. Hullermeier. Is an ordinal class structure useful in classifier learning? *International Journal of Data Mining, Modelling and Management*, 1(1):45–67, 2008.

[11] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.

[12] C. Li, H. Liu, and Z. Ming. Empirical study on virtual order class labels in nominal classification. *International Journal of Machine Learning and Cybernetics*, 13(11):3255 – 3266, 2022.

[13] G. Martínez-Muñoz and A. Suárez. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 609–616, New York, NY, USA, 2006. Association for Computing Machinery.

[14] R. Mohapatra, S. Saha, and S. S. Dhavala. Adaswarm: A novel pso optimization method for the mathematical equivalence of error gradients. *arXiv preprint arXiv:2006.09875*, 2020.

[15] S.-H. Park and J. Fürnkranz. Efficient prediction algorithms for binary decomposition techniques. *Data Mining and Knowledge Discovery*, 24(1):40–77, 2012.

[16] 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*, 25(2):289–302, 2014.

[17] P. Schiilkop, C. Burgest, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the 1st International Conference on Knowledge Discovery & Data Mining*, pages 252–257, 1995.

[18] J. Sánchez-Monedero, P. Gutiérrez, P. Tiňo, and C. Hervás-Martínez. Exploitation of pairwise class distances for ordinal classification. *Neural Computation*, 2013.

[19] K. Tumer and J. Ghosh. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29(2):341–348, 1996.

[20] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3-4):385–403, 1996.

[21] B. Vega-Márquez, I. A. Nepomuceno-Chamorro, C. Rubio-Escudero, and J. C. Riquelme. Ocean: Ordinal classification with an ensemble approach. *Information Sciences*, 580(1109/72), 2021.

[22] A. Xny, A. Khl, and B. Stl. A ternary bitwise calculator based genetic algorithm for improving error correcting output codes - sciencedirect. *Information Sciences*, 537:485–510, 2020.

[23] Y. Yang, B. Chen, Z. Yang, and D. Steinley. An algorithm for ordinal classification based on pairwise comparison. *Journal of Classification*, 37(1):158–179, 2020.

[24] J. T. Zhou, I. W. Tsang, S.-S. Ho, and K.-R. Muller. N-ary decomposition for multi-class classification. *Machine Learning*, 108:809–830, 2019.