Evolutionary Classifier Chain for Multi-Dimensional Classification

Yu-Yang Zhang[1,3], **Bin-Bin Jia**[2], **Min-Ling Zhang**[1,3*]

[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China
[3]Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
zhangyyang@seu.edu.cn, jiabinbin@lut.edu.cn, zhangml@seu.edu.cn* (corresponding author)

Abstract

In multi-dimensional classification (MDC), the classifier chain approach is based on a chain structure to model dependencies between class spaces. However, current research on constructing a chain order is usually based on a greedy criterion or random generation, which is highly likely to lead to an incorrect chain order and fit incorrect class dependencies. Moreover, existing classifier chain-based approaches do not consider the misleading effects of irrelevant input features on the classifiers. To fill the above gap, a classifier chain-based approach incorporating evolutionary chain order optimization and feature selection (ECCO) is proposed. Specifically, this approach designs a meta-heuristic algorithm to optimize the chain order of multiple classifiers. Simultaneously, the approach selects dimension-specific feature combinations that are more conducive to class prediction of each dimension. These strategies enhance the class prediction capability of the constructed MDC model. Comparative experiments on 14 real datasets validate that ECCO outperforms 7 state-of-the-art MDC approaches.

Introduction

Multi-dimensional Classification (MDC) refers to situations in data analysis and machine learning where a single sample may simultaneously be associated with multiple class variables (Van Der Gaag, De Waal et al. 2006; Zaragoza et al. 2011). Each class variable corresponds to one class space. This problem is characterized by the fact that these class spaces characterize the different semantics of the sample in multiple dimensions (Gil-Begue, Bielza, and Larrañaga 2021). MDC problems are very common in many practical applications, such as text classification and image annotation. In news article classification (Theeramunkong and Lertnattee 2002), an article about a technology company expanding its business in the global market can be labeled under the "*topic*" dimension as "*technology*", "*politics*" or "*business*". In the "*sentiment*" dimension, it can be labeled as "*negative*", "*neutral*" or "*positive*". Therefore, it needs to be assigned multiple labels with different dimensions simultaneously to reflect its diverse content when classifying the article. In image recognition (Wang et al. 2024), a street scene image can be labeled under the "*object*" dimension as "*pedestrian*", "*building*" or "*car*". In the "*time*" dimension, it can be labeled as "*daytime*", "*evening*" or "*late night*". Consequently, more precise image recognition techniques need to correctly identify labels in all dimensions of an image. These examples demonstrate the application and importance of MDC problems in various fields.

In general, the output space of an MDC problem is $\mathcal{Y} = C_1 \times C_2 \times ... \times C_q$, which corresponds to the Cartesian product of q class spaces. In the j-th class space ($1 \leq j \leq q$), the class space C_j containing K_j class labels can be expressed as $C_j = \{c_1^j, c_2^j, ..., c_{K_j}^j\}$. In contrast to traditional multi-class classification problems (Ou and Murphey 2007; Jia et al. 2023), the difficulty of the MDC problem lies in modeling the complex dependencies between many classes to achieve accurate labeling of all labels in multiple class dimensions. Existing approaches for dealing with multi-dimensional classification problems have been classified into three main categories: intuitive algorithms, explicit dependency-modeling algorithms and implicit dependency-modeling algorithms (Jia and Zhang 2023). Binary Relevance (BR) and Class Powerset (CP) approaches both belong to the intuitive algorithms. Their classification results are usually poor because the former handles class dependencies too simply and the latter overfits the class dependencies. Explicit dependency-modeling algorithms are represented by the Classifier Chain (CC) approaches (Read et al. 2021). This approach fits class dependencies by establishing a chaining order of classifiers. However, most of the existing CC-based approaches (Read, Martino, and Luengo 2014; Jia and Zhang 2022a) are based on greedy criteria or randomly generated chain ordering, which usually fails to accurately fit class dependencies. Implicit dependency-modeling algorithms are represented by feature augmentation (Jia and Zhang 2020) and label encoding (Liu et al. 2024; Jia and Zhang 2021; Tang et al. 2024). Such approaches mainly try to manipulate the feature space or transform the label space to alleviate the challenges posed by heterogeneity between classes. Nevertheless, these methods fail to give the user an explicit representation of class dependencies and lead to additional computational costs.

Several recent studies have shown the potential validity of manipulating the feature space implicitly to model dependencies between classes (Jia and Zhang 2022c). This is be-

*Corresponding authors

cause augmented features related to each class space can be initially generated based on samples that are close together in the output space. This useful discriminative information aids in subsequent label recognition across multiple dimensions. However, existing methods utilize all input features to model, but ignore the interference of redundant and irrelevant features for the identification of labels for each dimension. Therefore, this paper is conceptualized based on the following assumption. When dealing with the MDC problem, it is more reasonable to create a dimension-specific feature combination for each class space. This better captures the correlation between the feature space and the semantics of the heterogeneous labels of class variables.

To address the above problems, we propose a novel MDC approach named ECCO (i.e., *Evolutionary Classifier Chain Order*) that incorporates evolutionary chain order optimization and feature selection. This approach considers both the output class space and the input feature space for dependency modeling. While globally optimizing the prediction order of class spaces, the dimension-specific feature combination is designed for each class space. In this way, the classifier based on dimension-specific features can achieve more accurate label prediction in each dimension. Our contribution can be summarized as follows:

- Completely different from previous CC approaches, the proposed ECCO introduces evolutionary computation for the first time. The optimal chain order of the classifiers is found based on the powerful search capability of population evolution in a large search space.
- We are the first to introduce evolutionary feature selection into the MDC algorithm by constructing the dimension-specific feature combination for each class space.
- The approach we devised considers modeling in both the input feature space and the output class space, which provides a novel paradigm for solving the MDC problem.
- Based on the results of Wilcoxon and Friedman tests, ECCO ranks first across all three metrics.

The remainder of this paper is organized as follows. The next section describes the works with two modeling methods on MDC. After that, a detailed description of our proposed ECCO is presented. Then, the experimental setup and analysis of the experimental results are shown. Finally, the paper is summarized.

Related Work

The key to solving the MDC problem lies in modeling the class dependencies. Existing research interests focus on two main categories of explicit and implicit dependency modeling (Jia and Zhang 2024; Shi et al. 2025).

Explicit Dependency-Modeling Algorithms

Existing algorithms based on explicit dependency modeling aim to explicitly model dependencies by directly manipulating the category space through some structure (Read et al. 2011). Jia et al. (2022a) proposed a decomposition-based CC approach to solve the MDC problem by transforming it into multiple binary classification problems. However, this integration method can lead to an expensive computation and does not eliminate the impact of the chain order with mismatched dependencies. Jia et al. (2022b) also proposed a maximum margin MDC algorithm. The approach first maximizes the margin between each pair of class labels by one-versus-one decomposition. Then the modeling of class dependencies is achieved by the covariance regularization technique.

Another popular algorithm is to divide the class space into multiple super-classes (Read, Bielza, and Larrañaga 2014). This approach aims to divide the grouping by measuring certain metric conditions between class variables. However, it is difficult to include the full combinations of class spaces in a training set.

Implicit Dependency-Modeling Algorithms

The implicit dependency modeling algorithm aims to transform the MDC problem into a new problem without directly manipulating the class space. There are three popular research directions for this approach: label space transformation, sparse label encoding and feature augmentation.

For the first direction, the basic idea of one-hot multi-dimensional transformation is to convert the multi-dimensional output space into a binary-valued output space by one-vs-rest decomposition (Ma and Chen 2018). This approach makes the transformed problem suitable for solution by well-established multi-label classification methods. However, this approach tends to result in an inflated label space and ignores the hierarchical relationships between class variables.

For the second direction, SLEM (Jia and Zhang 2021) maps the original output space to a real-valued label space using a random encoding matrix. Then, a multi-output regression model is trained in the resulting label space to identify unseen examples. Ahmed et al (2023) proposed an end-to-end MDC approach. A high-performance deep learning classifier is realized by building a hypercube classifier and multiple DSOC neural networks connected to the hypercube. Zhang et al. (2022) proposed an end-to-end model of an adversarial variational autoencoder with regularized streams, which encodes both features and class variables into the same probabilistic space. However, the encoding and mapping methods in the above approaches often need to be tuned. Otherwise, the robustness and generalization performance of the model may be insufficient.

For the third direction, augmented features are generated by learning to enrich the discriminative information in the input space to aid in the classification of subsequent models. Jia et al. (2020) acquired augmented features by statistical information of each class space based on the K-nearest neighbor technique. LEFA (Wang et al. 2020) aims to generate augmented features by the deep learning technique. However, the above approaches do not fit the dependencies between class variables such as prediction order well. And they do not consider the interaction between input features.

The ECCO Approach

Existing CC-based approaches are usually not efficiently optimized for the chain order. In addition, current MDC ap-

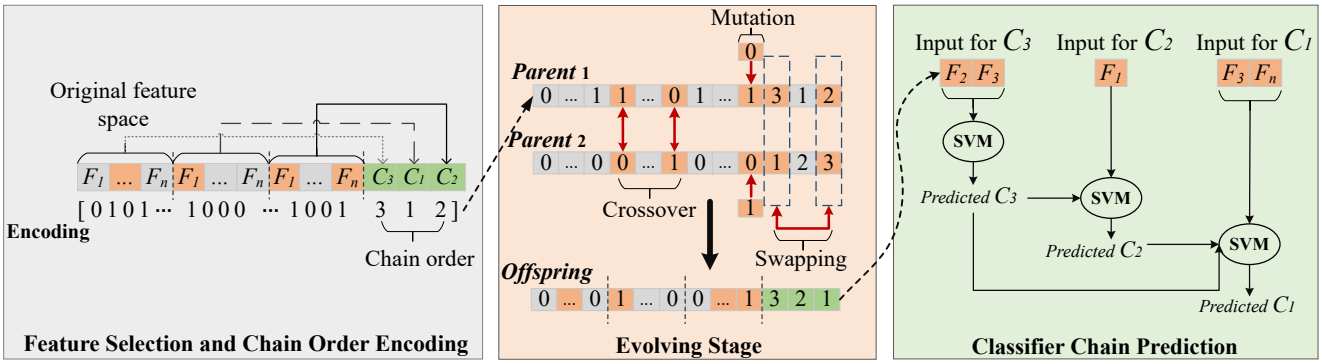

Figure 1: The workflow of the proposed ECCO approach when the number of class dimensions is 3.

proaches rarely consider the construction and selection of effective dimension-specific feature combinations. Therefore, we design a unified encoding optimization framework based on evolutionary algorithms to optimize both chain order and feature combinations. As shown in Figure 1, our framework is divided into three main parts. First, a hybrid encoding method is designed for issues to be optimized. Second, inspired by DDFS (Liang et al. 2024a), a two-population coevolutionary genetic algorithm (GA) is improved based on the problem characteristics of MDC to avoid falling into local optimum. Third, the classification performance is tested on the chain order and features that have evolved iteratively. Notably, the notations in Algorithms 1 and 2 are explained in detail in the **Supplementary Material**[1].

Encoding

In CC-based approaches, the input feature combination of each classifier has a coupling relationship with the chain order of the classifiers, both of which affect the classification accuracy. Therefore, we propose a hybrid encoding method to simultaneously perform evolutionary chain order optimization and dimension-specific feature selection. Without loss of generality, we will discuss the proposed approach when the number of class dimensions q is 3 as shown in Figure 1. First, three original feature spaces are merged with the chain order vector into a single individual. The selection or non-selection of each feature is denoted by 1 or 0. The chain order is represented using a vector like $(3, 1, 2)$. ECCO performs evolution by two subpopulations $\mathbf{P}_N$ and $\mathbf{P}_M$ consisting of these encoding vectors. Then, the exploration of dimension-specific feature combinations and chain order spaces is realized by performing different variation operations on $\mathbf{P}_N$ and $\mathbf{P}_M$. Finally, each offspring is decoded into the dimension-specific features for each space as well as the chain order. And the classification effect of each offspring is evaluated by sequentially feeding dimension-specific features and the predicted labels of previous classifiers into the current classifier according to the chain order.

[1]Code package and supplementary material are publicly available at: http://palm.seu.edu.cn/zhangml.

Evolving Stage

In ECCO, genetic operators are crucial for generating new individuals from the parent population. Algorithm 1 specifically demonstrates swapping, crossover and mutation, respectively. In both subpopulations $\mathbf{P}_N$, $\mathbf{P}_M$, the swapping and mutation operations are the same while the crossover operation is different.

Algorithm 1: Genetic Operators in ECCO

Input: N: overall population size; $\mathbf{P}_N$, $\mathbf{P}_M$: the subpopulation based on dominance or decomposition; d: original feature dimension; q: dimension of class spaces.
Output: $\mathbf{O}$: the overall offspring population.

1: Initialize parameters: $proC = 1$, $proM = 1$;
2: **Swapping in $\mathbf{P}_N$ or $\mathbf{P}_M$:**
3: Randomly select two class dimensions q_1 and q_2 out of q dimensions;
4: Swap the column $q * d + q_1$ with the column $q * d + q_2$ for $\mathbf{P}_N$ or $\mathbf{P}_M$;
5: **Crossover of GA Operators in $\mathbf{P}_N$:**
6: Split $\mathbf{P}_N(:, 1 : q * d)$ into $\mathbf{P}_1$ and $\mathbf{P}_2$ with an equal number of rows;
7: $N_1 \leftarrow$ the number of rows in $\mathbf{P}_1$;
8: $\mathbf{K} \leftarrow$ a logic matrix of size $(N_1, q * d)$ with elements set to 0 when generated random values are less than 0.5;
9: $\mathbf{K}(\text{repmat}(\text{rand}(N_1, 1) > proC, 1, q * d)) \leftarrow$ false;
10: $\mathbf{O}_1 = \mathbf{P}_1$; $\mathbf{O}_2 = \mathbf{P}_2$;
11: $\mathbf{O}_1(\mathbf{K}) = \mathbf{P}_2(\mathbf{K})$; $\mathbf{O}_2(\mathbf{K}) = \mathbf{P}_1(\mathbf{K})$;
12: $\mathbf{O}_N = [\mathbf{O}_1; \mathbf{O}_2]$;
13: **Crossover of GA-half Operators in $\mathbf{P}_M$:**
14: Split $\mathbf{P}_M(:, 1 : q * d)$ into $\mathbf{P}_1$ and $\mathbf{P}_2$ with an equal number of rows;
15: $\mathbf{K}$ is obtained according to lines 7-9 of Algorithm 1;
16: $\mathbf{O}_M = \mathbf{P}_1$;
17: $\mathbf{O}_M(\mathbf{K}) = \mathbf{P}_2(\mathbf{K})$;
18: **Mutation in $\mathbf{P}_N$ or $\mathbf{P}_M$:**
19: $\mathbf{S} \leftarrow$ a logic matrix of size $(N/2, q * d)$ with random values less than $\frac{proM}{q*d}$;
20: $\mathbf{O}_N(\mathbf{S}) = \sim \mathbf{O}_N(\mathbf{S})$; $\mathbf{O}_M(\mathbf{S}) = \sim \mathbf{O}_M(\mathbf{S})$;
21: $\mathbf{O} = [\mathbf{O}_N; \mathbf{O}_M]$
22: **return** $\mathbf{O}$

To fully explore the chain order space, we designed to randomly pick two dimensions of class spaces to swap in the parent matrix at each generation. As shown in the second stage in Figure 1, the original chain order in each solution is changed. This can be effective in exploring the class space with higher dimensions in the MDC problem.

For crossover, the parent population $\mathbf{P}_N$ ($\mathbf{P}_M$) is divided equally into two groups $\mathbf{P}_1$ and $\mathbf{P}_2$. This matrix $\mathbf{K}$ denotes the position vectors of the features that perform the crossover operation. For the GA crossover, the elements of $\mathbf{O}_1$ are replaced by the corresponding elements of $\mathbf{P}_2$ when the values of the corresponding positions in $\mathbf{K}$ are 1. And it is similar to $\mathbf{O}_2$. The final offspring is the combination of $\mathbf{O}_1$ and $\mathbf{O}_2$. For the GA-half crossover, the elements of $\mathbf{P}_1$ are only replaced by the corresponding elements of $\mathbf{P}_2$ where the elements in $\mathbf{K}$ are 1.

For mutation, a binary logic matrix $\mathbf{S}$ of size $(N/2, q{*}d)$ is created with values less than $\frac{proM}{q*d}$, indicating the positions to be mutated. The elements at these positions are flipped to guide the subpopulation to explore new feature areas. This combination of crossover and mutation operators ensures diversity of generated feature combinations and aids in the effective exploration of the solution space.

Label Prediction and Evaluation

To evaluate the performance of each of the generated solutions on the MDC problem, we design f_1 and f_2 to be computed. As shown in Eq. (1), f_1 and f_2 are simultaneously optimized to achieve a win-win situation in terms of classification performance and cost. $Hamming\ score$ (HS) is computed based on Eq. (4) to evaluate the MDC performance of each solution.

$$\text{Minimize} \begin{cases} f_1: \ Ratio\ of\ selected\ features \\ f_2: \ 1 - Hamming\ Score \end{cases} \tag{1}$$

The objective f_1 is calculated by the number of selected features, which is 5, dividing d in the third part of Figure 1. The objective f_2 can be calculated as follows. The third part of Figure 1 shows the classifier chain prediction process with $q = 3$. First, the obtained individual is decoded. In Figure 1, the optimized chain order is $(3, 2, 1)$. Therefore, dimensional intervals $[1, d]$, $[d + 1, 2d]$ and $[2d + 1, 3d]$ of the encoding are represented as the input features selected for class spaces C_3, C_2 and C_1, respectively. $\{F_2, F_3\}$ is the input feature combination of C_3. According to the linking rules of the classifier chain, the predicted labels obtained on C_3 with the feature F_1 are given as input to the classifier of C_2. The features $\{F_3, F_n\}$ and the predicted labels obtained on C_3 and C_2 are given as input to the classifier of C_1. Finally, the performance indicator HS for each solution can be computed based on the predicted labels and the ground-truth labels in all class spaces.

Overall Algorithmic Framework

In order to show our designed algorithm in more detail, the specific flow of ECCO is described in Algorithm 2. First, we designs two subpopulations $\mathbf{P}_N, \mathbf{P}_M$ to help each other in the evolutionary process. This allows for a better balance between exploration and exploitation. In the initial population, the selected features and chain order are randomly generated. Second, crossover and mutation are performed on the feature selection part by the GA operator as shown in the evolving stage in Figure 1. This can quickly find promising solutions in the huge search space of 2^{q*d}. A swapping strategy is designed for evolutionary chain order optimization to ensure that the space of possible chain orders is adequately searched. Third, two objectives are computed after each solution is decoded into the dimension-specific input features of each dimension as well as the chain order. Notably, we consider the tradeoff between the number of selected features and the classification effectiveness as shown in Eq. (2) and Eq. (3).

Algorithm 2: Overall Framework of ECCO

Input: Training set; test set; N: overall population size.
Output: The encoding of solution $\boldsymbol{x}_{Best}$ and its f_1 and f_2.

$\mathbf{P}_N \leftarrow$ Initialize $N/2$ solutions for the dominance-based subpopulation;
$\mathbf{P}_M \leftarrow$ Initialize $N/2$ solutions for the decomposition-based subpopulation;
Evaluate the objective values of $\mathbf{P}_N, \mathbf{P}_M$ respectively on the training set according to Eq. (1);
while the termination criterion is not met **do**
  $\mathbf{O}_N, \mathbf{O}_M \leftarrow$ Offspring generation based on parents of $\mathbf{P}_N, \mathbf{P}_M$ by the GA operator respectively;
  Decode and evaluate the objective values of $\mathbf{P}_N, \mathbf{P}_M$ on the training set according to Eq. (1);
  Update the ideal point Z^*;
  $\mathbf{P}'_N \leftarrow$ Perform environmental selection in $\mathbf{P}_N \cup \mathbf{O}_N$;
  $\mathbf{P}'_M \leftarrow$ Perform environmental selection in $\mathbf{P}_M \cup \mathbf{O}_M$;
  $\mathbf{P}_A = \mathbf{P}'_N \cup \mathbf{P}'_M$
  $\boldsymbol{v}_{Index} \leftarrow$ Perform non-dominated sorting on $\mathbf{P}_A$ based on their objective values;
  $\mathbf{P}''_M = \mathbf{P}_A(\boldsymbol{v}_{Index}(1 : N/2))$;
  $\mathbf{P}''_N = \mathbf{P}_A \setminus \mathbf{P}''_M$;
end while
$\mathbf{P}'_A = \mathbf{P}''_N \cup \mathbf{P}''_M$
Decode and evaluate the objective values of $\mathbf{P}'_A$ on the test set according to Eq. (1);
Perform a non-dominated ordering of the obtained objective values to find the solution $\boldsymbol{x}_{Best}$;
return The encoding of $\boldsymbol{x}_{Best}$ and its f_1 and f_2.

In the environmental selection phase, ECCO can select solutions with as few features as possible but better classification results. After all parent and offspring solutions have been evaluated on f_1 and f_2, environmental selection is performed to obtain the population for the next generation. Specifically, the Pareto dominance relationship (Deb et al. 2002) is used to select the new population $\mathbf{P}'_N$. If $\boldsymbol{b}$ is not worse than $\boldsymbol{a}$ in both objective values f_1 and f_2 defined in Eq. (1), then $\boldsymbol{b}$ is given a smaller front number than $\boldsymbol{a}$. The discriminant condition is shown as:

$$\begin{aligned} &\forall i : f_i(\boldsymbol{b}) \leq f_i(\boldsymbol{a}) \wedge \exists j : f_j(\boldsymbol{b}) < f_j(\boldsymbol{a}) \\ &\text{s.t.} \quad \boldsymbol{a}, \boldsymbol{b} \in \Omega \end{aligned} \tag{2}$$

The $N/2$ solutions with smaller front numbers are selected

into $\mathbf{P}'_N$. The Tchebycheff method (Zhang and Li 2007) is used to select individuals with better convergence in $\mathbf{P}_M$ and $\mathbf{O}_M$. On i-th weight vector, the values of the aggregation function of the parent and child are similarly calculated as

$$\begin{aligned} \min \quad & g(\boldsymbol{u} \mid \boldsymbol{\lambda}, Z^*) = \max\left\{\boldsymbol{\lambda}_i \left(f_i(\boldsymbol{u}) - Z^*\right)\right\} \\ \text{s.t.} \quad & \boldsymbol{u} \in \Omega \end{aligned} \tag{3}$$

where g is the aggregation function, $\boldsymbol{u}$ is the encoding vector and $\boldsymbol{\lambda}$ is the weight vector. Z^* is the ideal point, whose two components are the minimum values of the entire population on f_1 and f_2 defined in Eq. (1), respectively. And the solution with the smaller value of g that is also closer to the ideal point is retained. The selected $\mathbf{P}'_N$ and $\mathbf{P}'_M$ are merged into a single population $\mathbf{P}_A$, which combines good diversity and convergence. Finally, individuals in $\mathbf{P}_A$ are reclassified into two subpopulations $\mathbf{P}''_N, \mathbf{P}''_M$ based on their evolutionary status $\boldsymbol{v}_{Index}$. The transfer of search experience between the two evolutionary forms can accelerate the convergence of the algorithm.

The algorithm iteratively optimizes the population of solutions until the maximum number of evaluations is reached. The final generation of solutions is output to a test set to examine their performance metrics on an MDC problem. Finally, the solution $\boldsymbol{x}_{Best}$ on the Pareto front with the best performance on f_2 is the output of ECCO.

Experiments

In this section, we conduct comparative experiments with 7 comparison algorithms on 14 real datasets. All experiments are executed with 50 parallel cores invoked on the same workstation with two Intel(R) Xeon(R) Gold 6230 CPUs.

Datasets

In order to fully validate the effectiveness of the proposed algorithm, we selected 14 real datasets to examine the classification performance of our algorithm on the MDC problem. The number of examples (#Examples), the number of class dimensions (#Dim.), the number of labels in each class space (#Labels/Dim.) and the number of features (#Features) for each dataset are shown in Table 1. It can be seen that some of the datasets with many class spaces have a high number of input features (i.e., Oes97, Oes10, WaterQuality, Rf1 and Pain). Therefore, the experimental datasets can fully validate the effectiveness of the algorithm on the MDC problem.

DataSet	#Examples	#Dim.	#Labels/Dim.	#Features
Edm	154	2	3	16n
Flare1	323	3	3,4,2	10x
Oes97	334	16	3	263n
Jura	359	2	4,5	9n
Oes10	403	16	3	298n
Enb	768	2	2,4	6n
Song	785	3	3	98n
WQplants	1060	7	4	16n
WQanimals	1060	7	4	16n
WaterQuality	1060	14	4	16n
BeLaE	1930	5	5	1n,44x
Voice	3136	2	4,2	19n
Rf1	8987	8	4,4,3,4,4,3,4,3	64n
Pain	9734	10	2,5,4,2,2,5,2,5,2,2	136n

Table 1: Basic information of the experimental dataset

Compared Approaches

In this paper, seven state-of-the-art MDC approaches are used to compare with the proposed ECCO. They are described carefully as follows:

- BR (Zhang et al. 2018) solves the MDC problem by independently training multiple multi-class classifiers.
- CP (Tsoumakas, Katakis, and Vlahavas 2011) transforms the MDC problem into a single multi-class classification problem.
- ECC (Read et al. 2011) solves the MDC problem by training multiple chains of classifiers to integrate and vote. The chain order of the classifiers is generated based on random reordering.
- BCC (Zaragoza et al. 2011) builds multiple classifier chains based on the Bayesian network for ensemble learning.
- EDCC (Jia and Zhang 2022a) overcomes the problem that classifier chains are prone to propagate errors by one-versus-one decomposition strategy and ensemble classifier chains.
- gMML (Ma and Chen 2018) is based on a newly designed label transformation method that aims to minimize the Mahalanobis distance between the regression output and the true label vectors.
- SEEM (Jia and Zhang 2020) models class dependencies through a two-layer strategy, which is based on CP and BR respectively.

Specifically, the aforementioned comparison algorithms can be categorized into three groups. BR and CP are basic algorithms. ECC, BCC and EDCC are three advanced algorithms based on classifier chains. gMML and SEEM are advanced algorithms based on implicit and explicit modeling, respectively.

Parameter Settings

For each of the seven compared algorithms, the parameters in the code are consistent with the version in the corresponding paper respectively. We use a Support Vector Machine (SVM) as the base classifier for each comparison algorithm. For the proposed algorithm ECCO, the population size is empirically set to 200. And the number of iterations is set to 100 to balance diversity and computational efficiency (Liang et al. 2024a). For the fairness and validity of the experimental results, the base classifier of ECCO is SVM.

In the experimental phase, each algorithm is run 30 times and the results are averaged to rule out randomness. Each dataset is partitioned into the training set and the test set in the ratio of 80% and 20%. It is worth noting that this division is kept constant for different comparison algorithms on the same dataset. This ensures the fairness of the experiments. In the training phase, 5-fold cross-validation is performed to

DataSets	*Hamming Score*							
	ECCO	BR	CP	ECC	BCC	EDCC	gMML	SEEM
Edm	**0.738**	0.672	0.641	0.641	0.734	0.672	0.672	0.625
Flare1	**0.904**	**0.904**	**0.904**	0.889	**0.904**	0.899	**0.904**	**0.904**
Oes97	**0.723**	0.720	0.640	0.714	0.660	0.717	0.699	0.715
Jura	0.641	0.628	0.608	**0.649**	0.561	0.642	0.622	0.547
Oes10	0.786	0.786	0.705	0.781	0.742	0.784	0.748	**0.790**
Enb	**0.790**	0.750	0.740	0.747	0.779	0.503	0.740	0.711
Song	**0.793**	0.789	0.732	**0.793**	0.656	0.776	0.776	0.785
WQplants	**0.706**	0.702	0.478	0.703	0.606	0.700	0.702	0.703
WQanimals	**0.663**	0.648	0.456	0.654	0.545	0.655	0.659	0.662
WaterQuality	**0.692**	0.677	0.535	0.688	0.579	0.687	0.689	0.688
BeLaE	**0.481**	0.371	0.320	0.416	0.381	0.451	0.425	0.441
Voice	0.940	0.845	0.774	0.787	**0.961**	0.843	0.687	0.821
Rf1	**0.794**	0.627	0.600	0.612	0.606	0.736	0.550	0.712
Pain	**0.960**	0.959	**0.960**	0.959	0.942	**0.960**	0.958	0.959
Friedman's rank	**1.679**	4.107	6.607	4.500	5.607	4.179	5.071	4.179

Table 2: Mean results of ECCO and 7 comparison algorithms on HS.

DataSets	*Exact Match*							
	ECCO	BR	CP	ECC	BCC	EDCC	gMML	SEEM
Edm	0.481	0.406	0.438	0.406	**0.500**	0.406	0.438	0.406
Flare1	**0.773**	**0.773**	**0.773**	0.727	**0.773**	0.758	**0.773**	**0.773**
Oes97	0.013	**0.015**	**0.015**	**0.015**	0.000	**0.015**	**0.015**	**0.015**
Jura	**0.411**	0.405	0.378	0.378	0.351	0.405	0.378	0.338
Oes10	0.049	0.037	0.049	0.037	**0.061**	0.049	0.037	0.049
Enb	**0.581**	0.500	0.481	0.493	0.559	0.513	0.481	0.506
Song	0.501	0.500	0.430	**0.525**	0.278	0.487	0.500	0.487
WQplants	0.172	0.169	0.019	**0.174**	0.066	0.164	0.169	0.164
WQanimals	0.096	0.061	0.000	0.079	0.009	0.079	0.093	**0.098**
WaterQuality	**0.031**	0.023	0.000	0.028	0.000	0.019	0.028	0.028
BeLaE	**0.055**	0.013	0.008	0.028	0.018	0.028	0.021	0.028
Voice	0.883	0.706	0.574	0.590	**0.922**	0.700	0.486	0.653
Rf1	**0.190**	0.004	0.081	0.043	0.024	0.118	0.000	0.098
Pain	**0.798**	0.795	**0.798**	0.795	0.701	0.794	0.796	0.785
Friedman's rank	**2.250**	4.821	5.536	4.607	5.214	4.464	4.643	4.464

Table 3: Mean results of ECCO and 7 comparison algorithms on EM.

DataSets	*Sub-Exact Match*							
	ECCO	BR	CP	ECC	BCC	EDCC	gMML	SEEM
Edm	**0.994**	0.938	0.844	0.875	0.969	0.938	0.906	0.844
Flare1	**0.939**	**0.939**	**0.939**	**0.939**	**0.939**	**0.939**	**0.939**	**0.939**
Oes97	0.073	0.075	0.045	**0.104**	0.045	**0.104**	0.060	0.060
Jura	0.872	0.851	0.838	**0.919**	0.770	0.878	0.865	0.757
Oes10	0.183	0.171	0.122	0.146	0.134	0.171	0.146	**0.195**
Enb	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Song	**0.886**	0.873	0.797	0.861	0.741	0.848	0.835	0.873
WQplants	**0.400**	0.399	0.066	0.399	0.230	0.390	0.394	0.366
WQanimals	**0.281**	0.252	0.042	0.262	0.107	0.257	0.262	0.271
WaterQuality	0.110	0.061	0.000	0.103	0.009	0.108	**0.113**	0.099
BeLaE	**0.196**	0.106	0.049	0.126	0.085	0.155	0.147	0.144
Voice	0.997	0.984	0.975	0.984	**1.000**	0.987	0.887	0.989
Rf1	**0.442**	0.123	0.251	0.186	0.133	0.399	0.037	0.325
Pain	0.896	0.894	0.898	0.893	0.858	0.900	0.894	**0.901**
Friedman's rank	**2.286**	4.643	6.500	4.250	5.964	3.393	4.857	4.107

Table 4: Mean results of ECCO and 7 comparison algorithms on SEM.

Evaluation Metric	ECCO against						
	BR	CP	ECC	BCC	EDCC	gMML	SEEM
HS	**win**[1.12E-03]	**win**[1.23E-03]	**win**[3.85E-03]	**win**[3.07E-03]	**win**[1.51E-03]	**win**[3.07E-03]	**win**[2.90E-03]
EM	**win**[2.11E-03]	**win**[3.73E-03]	**win**[6.95E-03]	**win**[1.44E-02]	**win**[1.66E-03]	**win**[2.11E-02]	**win**[4.87E-03]
SEM	**win**[2.72E-03]	**win**[2.33E-03]	**win**[4.80E-02]	**win**[2.33E-03]	**win**[3.97E-02]	**win**[1.57E-02]	**win**[9.18E-03]

Table 5: Wilcoxon signed-ranks test results for ECCO and 7 comparison algorithms on 3 metrics

avoid bias in the evaluation of solutions on the validation set (Liang et al. 2024b).

Performance Metrics

Following the existing research on the MDC problem, we chose three metrics to evaluate the performance of all approaches. Given an MDC problem, the test set $\mathcal{S} = \{(\boldsymbol{x_i}, \boldsymbol{y_i}) \mid 1 \leq i \leq p\}$, where p is the number of test samples. l is the MDC model to be evaluated. q is the number of class spaces. y_{ij} is the ground-truth labels for the j-th dimension of the i-th test sample. $\hat{y}_{ij}$ is the predicted labels for the j-th dimension of the i-th test sample. Their detailed formulas are shown below:

- *Hamming Score* (HS) : This metric measures the average proportion of how many class spaces are predicted correctly for each test sample. It is calculated as follows:

$$\mathrm{HS}_{\mathcal{S}}(l) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{q}\cdot r^{(i)} \tag{4}$$

- *Exact Match* (EM) : This metric measures the average proportion of all class spaces predicted correctly for each test sample. It is calculated as follows:

$$\mathrm{EM}_{\mathcal{S}}(l) = \frac{1}{p}\sum_{i=1}^{p}[\![r^{(i)} = q]\!] \tag{5}$$

- *Sub-Exact Match* (SEM) : This metric is a relaxed version of EM designed to measure the average proportion of test samples with at least $q-1$ dimensions correctly classified. It is computed as follows:

$$\mathrm{SEM}_{\mathcal{S}}(l) = \frac{1}{p}\sum_{i=1}^{p}[\![r^{(i)} \geq q-1]\!] \tag{6}$$

Here, $r^{(i)} = \sum_{j=1}^{q}[\![y_{ij} = \hat{y}_{ij}]\!]$ denotes the number of correctly predicted dimensions.

Experimental Results

In this subsection, we count the average results of 30 runs of the proposed algorithm ECCO and the seven comparison algorithms on HS, EM and SEM as shown in Table 2, Table 3 and Table 4, respectively. In addition, the ranking results of the Friedman test for each algorithm are recorded in the last row of the tables. In order to explore the significance of the superiority of our approach, the Wilcoxon signed-ranks test at 0.05 significance level is performed. The results of the test are recorded in Table 5. Based on the experimental results recorded in the above table, the following observations can be observed:

- By evaluating and testing the three metrics, ECCO outperforms the seven comparison algorithms in 78.6%, 50% and 64.3% cases across all datasets, respectively. The results of the Friedman test showed that ECCO ranked the highest on all three metrics.
- ECCO is significantly better than BR and CP because of the latter two oversimplified considerations of class dependencies.
- ECCO significantly outperforms ECC, BCC and EDCC on most of the datasets. These CC-based approaches randomly or simply generate chain orders that do not correspond to real class dependencies. This can lead to instability in the constructed model. In addition, inputting all the original features for each classifier can also result in redundant and irrelevant features interfering with the classifiers. This indicates that constructing dimension-specific feature combinations and optimizing the chain order for each dimension of the classifier in ECCO is effective.
- ECCO also beats the advanced implicit dependency-modeling approaches gMML and SEEM on three metrics. This suggests that the existing approaches still suffer from the defect of insufficient input information when implicitly modeling. In contrast, ECCO simultaneously considers the optimization of the chain order and dimension-specific feature selection, which can provide more accurate and fuller recognition information for the classifier of each class space.

In addition, it is worth noting that we also conduct further analyses on the ablation experiments, parameter sensitivity, and computational complexity. Specific experimental results and analyses can be found in Section 3, 4, 5 and 6 of the **Supplementary Material**.

Conclusion

In this work, we propose for the first time an evolutionary optimization framework ECCO for the MDC problem that incorporates evolutionary chain order optimization and dimension-specific feature selection. This approach utilizes a hyper-heuristic algorithm to evolve the best solution by building a hybrid encoding of the two optimization problems. The superposition of useful information in the input and output spaces is realized to enhance the classification performance of the constructed MDC model. The experimental results demonstrate the effectiveness of ECCO in solving the MDC problem.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Science Foundation of China (62225602, 62306131) and the Big Data Computing Center of Southeast University.

References

Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2): 182–197.

Gil-Begue, S.; Bielza, C.; and Larrañaga, P. 2021. Multi-dimensional Bayesian network classifiers: A survey. *Artificial Intelligence Review*, 54(1): 519–559.

Jia, B.-B.; Liu, J.-Y.; Hang, J.-Y.; and Zhang, M.-L. 2023. Learning label-specific features for decomposition-based multi-class classification. *Frontiers of Computer Science*, 17(6): 176348.

Jia, B.-B.; and Zhang, M.-L. 2020. Multi-dimensional classification via kNN feature augmentation. *Pattern Recognition*, 106: 107423.

Jia, B.-B.; and Zhang, M.-L. 2021. Multi-dimensional classification via sparse label encoding. In *Proceedings of the 38th International Conference on Machine Learning*, 4917–4926.

Jia, B.-B.; and Zhang, M.-L. 2022a. Decomposition-based classifier chains for multi-dimensional classification. *IEEE Transactions on Artificial Intelligence*, 3(2): 176–191.

Jia, B.-B.; and Zhang, M.-L. 2022b. Maximum margin multi-dimensional classification. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12): 7185–7198.

Jia, B.-B.; and Zhang, M.-L. 2022c. Multi-dimensional classification via selective feature augmentation. *Machine Intelligence Research*, 19(1): 38–51.

Jia, B.-B.; and Zhang, M.-L. 2023. Multi-dimensional classification via decomposed label encoding. *IEEE Transactions on Knowledge and Data Engineering*, 35(2): 1844–1856.

Jia, B.-B.; and Zhang, M.-L. 2024. Multi-dimensional classification: paradigm, algorithms and beyond. *Vicinagearth*, 1(1): 3.

Liang, J.; Zhang, Y.; Chen, K.; Qu, B.; Yu, K.; Yue, C.; and Suganthan, P. N. 2024a. An evolutionary multiobjective method based on dominance and decomposition for feature selection in classification. *Science China Information Sciences*, 67(2): 120101.

Liang, J.; Zhang, Y.; Qu, B.; Chen, K.; Yu, K.; and Yue, C. 2024b. A multiform optimization framework for multiobjective feature selection in classification. *IEEE Transactions on Evolutionary Computation*, 28(4): 1024–1038.

Liu, X.; Zhu, J.; Tian, Z.; and Li, Z. 2024. Multi-dimensional classification via class space fusion and comprehensive label correlations. *Information Fusion*, 111: 102521.

Ma, Z.; and Chen, S. 2018. Multi-dimensional classification via a metric approach. *Neurocomputing*, 275: 1121–1131.

Ou, G.; and Murphey, Y. L. 2007. Multi-class pattern classification using neural networks. *Pattern Recognition*, 40(1): 4–18.

Read, J.; Bielza, C.; and Larrañaga, P. 2014. Multi-dimensional classification with super-classes. *IEEE Transactions on Knowledge and Data Engineering*, 26(7): 1720–1733.

Read, J.; Martino, L.; and Luengo, D. 2014. Efficient monte carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition*, 47(3): 1535–1546.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2011. Classifier chains for multi-label classification. *Machine Learning*, 85: 333–359.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2021. Classifier chains: A review and perspectives. *Journal of Artificial Intelligence Research*, 70: 683–718.

Saleh, A. A.; and Weigang, L. 2023. Deep self-organizing cube: A novel multi-dimensional classifier for multiple output learning. *Expert Systems with Applications*, 230: 120627.

Shi, Y.; Ye, H.; Man, D.; Han, X.; Zhan, D.; and Jiang, Y. 2025. Revisiting multi-dimensional classification from a dimension-wise perspective. *Frontiers of Computer Science*, 19(1): 191304.

Tang, J.; Chen, W.; Wang, K.; Zhang, Y.; and Liang, D. 2024. Probability-based label enhancement for multi-dimensional classification. *Information Sciences*, 653: 119790.

Theeramunkong, T.; and Lertnattee, V. 2002. Multi-dimensional text classification. In *Proceedings of the 19th International Conference on Computational Linguistics*, 1–7.

Tsoumakas, G.; Katakis, I.; and Vlahavas, I. 2011. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7): 1079–1089.

Van Der Gaag, L. C.; De Waal, P. R.; et al. 2006. Multi-dimensional Bayesian network classifiers. In *Proceedings of the 3rd European Workshop on Probabilistic Graphical Models*, 107–114.

Wang, B.; Huang, G.; Li, H.; Chen, X.; Zhang, L.; and Gao, X. 2024. Hybrid CBAM-EfficientNetV2 fire image recognition method with label smoothing in detecting tiny targets. *Machine Intelligence Research*, 21(6): 1145–1161.

Wang, H.; Chen, C.; Liu, W.; Chen, K.; Hu, T.; and Chen, G. 2020. Incorporating label embedding and feature augmentation for multi-dimensional classification. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, volume 34, 6178–6185.

Zaragoza, J. H.; Sucar, L. E.; Morales, E. F.; Bielza, C.; and Larrañaga, P. 2011. Bayesian chain classifiers for multidimensional classification. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence*, volume 3, 2192–2197.

Zhang, M.-L.; Li, Y.-K.; Liu, X.-Y.; and Geng, X. 2018. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12(2): 191–202.

Zhang, Q.; and Li, H. 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6): 712–731.

Zhang, W.; Gou, Y.; Jiang, Y.; and Zhang, Y. 2022. Adversarial VAE with normalizing flows for multi-dimensional classification. In *Proceedings of the 5th Chinese Conference on Pattern Recognition and Computer Vision*, 205–219.