ORIGINAL ARTICLE

Class binarization to neuroevolution for multiclass classification

Gongjin Lan[1] · Zhenyu Gao[2] · Lingyao Tong[3] · Ting Liu[2]

Abstract

Multiclass classification is a fundamental and challenging task in machine learning. The existing techniques of multiclass classification can be categorized as (1) decomposition into binary (2) extension from binary and (3) hierarchical classification. Decomposing multiclass classification into a set of binary classifications that can be efficiently solved by using binary classifiers, called class binarization, which is a popular technique for multiclass classification. Neuroevolution, a general and powerful technique for evolving the structure and weights of neural networks, has been successfully applied to binary classification. In this paper, we apply class binarization techniques to a neuroevolution algorithm, NeuroEvolution of Augmenting Topologies (NEAT), that are used to generate neural networks for multiclass classification. We propose a new method that applies Error-Correcting Output Codes (ECOC) to design the class binarization strategies on the neuroevolution for multiclass classification. The ECOC strategies are compared with the class binarization strategies of One-vs-One and One-vs-All on three well-known datasets of *Digit*, *Satellite*, and *Ecoli*. We analyse their performance from four aspects of multiclass classification degradation, accuracy, evolutionary efficiency, and robustness. The results show that the NEAT with ECOC performs high accuracy with low variance. Specifically, it shows significant benefits in a flexible number of binary classifiers and strong robustness.

Keywords Multiclass classification · Binary classification · Error correcting output codes · NEAT · One-vs-one · One-vs-all

✉ Zhenyu Gao
  vu.gaozhy@gmail.com

✉ Ting Liu
  t.liu@vu.nl

  Gongjin Lan
  langj@sustech.edu.cn

  Lingyao Tong
  l.tong@vu.nl

1 Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China

2 Department of Computer Science, VU University Amsterdam, Amsterdam, The Netherlands

3 Department of Clinical, Neuro- & Developmental Psychology, VU University Amsterdam, Amsterdam, The Netherlands

1 Introduction

The classification tasks can be divided into binary (two-class) classification and multiclass classification. Multiclass classification is a crucial branch of machine learning, and has been applied in a wide variety of applications, such as medicine, speech recognition, and computer vision. The existing multiclass classification techniques can be basically divided into three categories, decomposition into binary, extension from binary, and hierarchical classification [1]. Although some classifiers such as Neural Networks (NNs) can classify multiple classes directly as a monolithic multiclass classifier, many state-of-the-art classifiers are inherently proposed for binary classification. Currently, a popular technique of multiclass classification is to decompose multiclass classification into binary classification [2], which is an efficient method to decode the classification, called class binarization. The class binarization approaches for multiclass classification have many advantages. First, developing binary classifiers is generally much easier than developing multiclass classifiers [3]. Second, many classifiers such as Support Vector Machine

(SVM) and C4.5 are inherently proposed for binary classification with outstanding performance [4, 5].

The binary classifiers (e.g., NNs and SVM) have been successfully applied to the decomposition of multiclass classification. Neural networks are generally designed by researchers manually. Using algorithms to automatically generate efficient neural networks is another popular approach for designing neural networks. Neuroevolution is a popular and powerful technique for evolving the structure and weights of neural networks automatically. Although neuroevolution approaches have been successfully applied to evolve efficient neural networks for binary classification, it generally struggles to generate neural networks for high accuracy in complex tasks such as multiclass classification [6]. In this work, we therefore investigate class binarization techniques in neuroevolution for multiclass classification.

NeuroEvolution of Augmenting Topologies (NEAT) is a popular neuroevolution algorithm that applies evolutionary algorithms (EAs) to generate desired neural networks by evolving both weights and structures [7]. NEAT-based approaches have been successfully applied to a broad range of machine learning tasks such as binary classification [8, 9], regression [10], and robotics [11]. However, it is notorious that neural networks evolved by NEAT-based approaches generally suffer severe multiclass classification degradation [6, 12]. The performance of neural networks evolved by NEAT degrades rapidly as the number of classes increases [6, 9]. To solve this issue, we apply the class binarization technique of Error-Correcting Output Codes (ECOC) to decompose multiclass classification into multiple binary classifications that NEAT-based approaches have been successfully applied to.

In general, there are three well-known types of class binarization approaches: One-vs-One (OvO), One-vs-All (OvA), and ECOC [2] (see Sect. 3.2). Theoretically, these three approaches work perfectly for multiclass classification when binary classifier predictions are 100% correct. However, realistic binary classifiers inevitably make wrong predictions, and these class binarization approaches therefore perform differently for multiclass classification. Although the class binarization techniques of OvO and OvA have been applied to NEAT-based multiclass classification [6], it is a novel method that applies ECOC to NEAT for multiclass classification, noted as ECOC-NEAT.

In this work, we mainly concentrate on the two research questions: (1) how ECOC-NEAT performs for multiclass classification? (2) how the size and quality of ECOC impact the performance of ECOC-NEAT for multiclass classification? To answer these two research questions, this study investigates (1) the performance of OvO-NEAT, OvA-NEAT, ECOC-NEAT, and the standard (original) NEAT for multiclass classification, (2) the performance of ECOC-NEAT with different number of classifiers and

different ECOCs. We analyse their performance from four aspects of multiclass degradation, accuracy, training efficiency, and robustness.

To the convincing conclusions, we choose three popular datasets, (*Digit*, *Satellite*, and *Ecoli*) that are usually used to evaluate the methods in multiclass classification. The main findings are summarized into two points.

(1) ECOC-NEAT offers various benefits compared to the standard NEAT and the NEAT with other class binarization techniques for multiclass classification.

- ECOC-NEAT performs comparable high accuracy as OvO-NEAT.
- ECOC-NEAT outperforms OvO-NEAT and OvA-NEAT in terms of robustness.
- ECOC-NEAT performs significant benefits in a flexible number of base classifiers.

(2) The size and quality of ECOC greatly influence the performance of ECOC-NEAT.

- Larger size ECOCs usually contribute to better performance for a given multiclass classification.
- High quality (optimized) ECOCs perform significantly better than normal ECOCs.

The rest of this paper is organized as follows. In Sect. 2, we provide an overview of the state-of-the-art studies of class binarization for multiclass classification. We present the methodology of NEAT and class binarization in Sect. 3. Datasets and experimental setup are addressed in Sect. 4. We present the results in Sect. 5 from four aspects: multiclass classification degradation, breadth evaluation, evolution efficiency, and robustness. Finally, we discuss this work in-depth and outlook the future work in Sect. 6, followed by the conclusions in Sect. 7.

2 Related work

OvO, OvA, and ECOC are three well-known class binarization techniques for multiclass classification. Although these three class binarization techniques have been successful applied to many applications, there is a lack of study that applies them (particularly ECOC) to neuroevolution for multiclass classification.

In [13], OvA is applied to the diagnosis of concurrent defects with binary classifiers of SVM and C4.5 decision tree. Adnan and Islam [14] applied OvA to the context of Random Forest. Allwein et al. proposed a general method for combining binary classifiers, in which the ECOC method is applied to a unifying approach with code matrices [15]. These studies applied the three class

binarization techniques into the traditional classifiers for multiclass classifications.

In the early studies of binary classification in neural networks and neuroevolution, Liu and Yao [16] proposed a new cooperative ensemble learning system for designing neural network ensembles, in which a problem is decomposed into smaller and specialized ones, and then each subproblem is solved by an individual neural network. Abbass et al. [17] and Garcia-Pedrajas et al. [18] presented evolution-based methods to design neural network ensembles. Lin and Damminda proposed a new algorithm of learning-NEAT that combines class binarization techniques and backpropagation for multiclass classification [8].

In the recent study [6], the class binarization techniques of OvO and OvA are applied to decompose multiclass multiclass into a set of binary classifications for solving the multiclass classification degradation of NEAT, in which binary classifiers are the individual NEAT-evolved neural networks. Two ensemble approaches of OvO-NEAT and OvA-NEAT are developed to achieve both higher accuracy and higher efficiency than the standard NEAT. Although the class binarization techniques of OvO and OvA have been applied to NEAT for multiclass classification, there is a lack of study that investigates the well-know technique of ECOC in NEAT for multiclass classification.

3 Methodology

In this section, we describe the neuroevolution algorithm of NEAT, the class binarization techniques of OvO, OvA, and ECOC.

3.1 NeuroEvolution of augmenting topologies

NEAT is a widely used neuroevolution algorithm that generates neural networks by evolving both weights and structure [7, 19]. NEAT evolves neural networks with flexible topology, starting from the elementary topology where all input nodes are connected to all output nodes, and adding nodes and connections via the operations of recombination and mutations, which leads to an augmented topology. In this work, NEAT is also allowed to delete nodes as well as connections. NEAT searches optimal neural networks through weight space and topological space simultaneously. There is no need for an initial or pre-defined fixed-topology that relies on the experience of researchers. Recombination and mutation induce an optimal topology of NN to an effective network.

An example of evolving neural networks by NEAT for multiclass classification is illustrated in Fig. 1. NEAT aims to generate an optimal neural network (i.e., highest fitness)
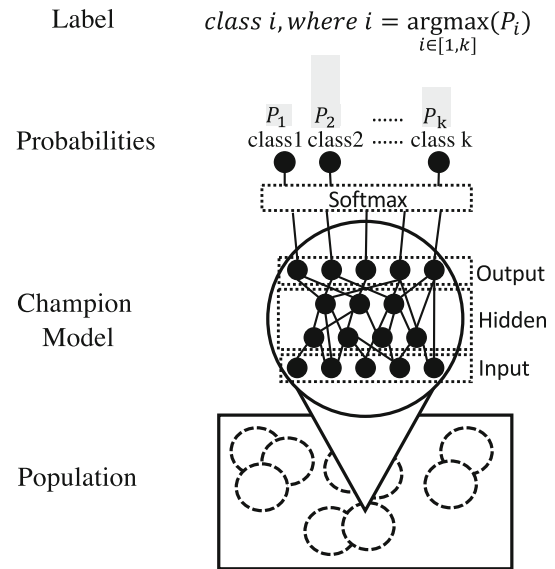

Fig. 1 Illustration of evolving neural networks by NEAT for multiclass classification

as the winning multiclass classifier. In particular, NEAT generates a binary classifier when the number of classes is two, where the NEAT is referred to as binary-NEAT (B-NEAT) as shown in the left part of Fig. 2. The number of nodes of the input layer is the dimensions of feature ($\mathcal{D}$), and the number of output nodes is the number of classes (k). We apply a softmax operation in the final layer to output probabilities of each class for multiclass classification. The class with the highest probability is predicted as the result.

NEAT is essentially a variant of evolutionary algorithms. Therefore, the fitness function is crucial to guide the convergence of evolving desired neural networks. In this work, we evaluate the performance of evolved neural networks with the prediction accuracy, that is the percentage of correct predictions. We note the number of correct predictions as $\mathcal{N}_c$, the number of total predictions as $\mathcal{N}_t$. The fitness (f) can be calculated as $f = \mathcal{N}_c / \mathcal{N}_t$.

Although NEAT can directly evolve neural networks for multiclass classification, it suffers the notorious multiclass classification degradation [8]. We apply NEAT as the baseline method for multiclass classification in this study, i.e., standard NEAT.

3.2 Class binarization

3.2.1 One-vs-One

The class binarization of OvO (also called All-vs-All) technique converts k-class classification into $\binom{k}{2}$ binary classifications that are constructed by using the class i ($i = $
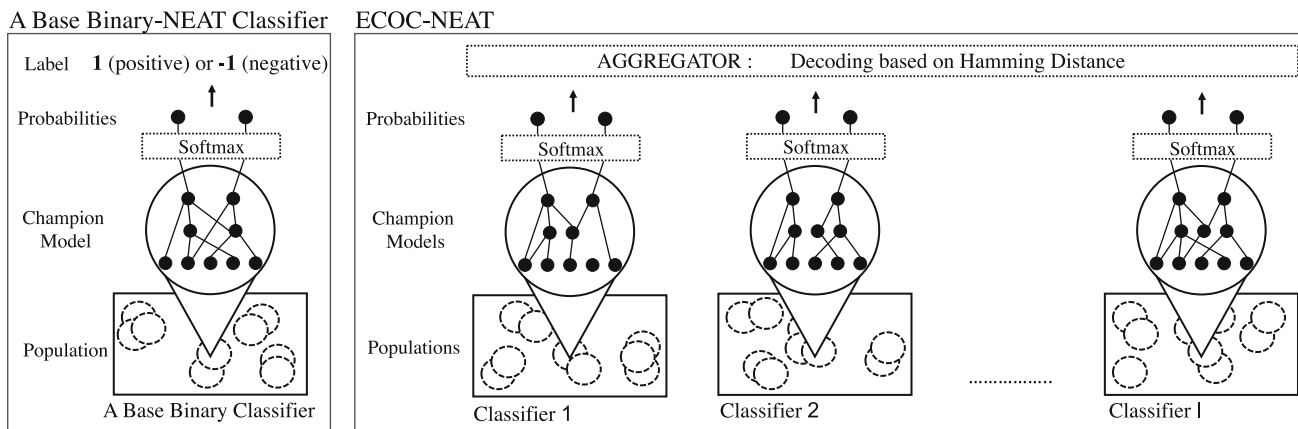
Fig. 2 ECOC-NEAT for multiclass classification. The left part shows an evolved base (binary) classifier by NEAT. The right part shows the ECOC-NEAT with base classifiers

$1, ..., k-1)$ as the positive examples and other classes $j > i$ $(j = 2, ..., k)$ as the negative examples [1]. That is, each class is compared with each other class separately. The existing studies [15, 20] show that OvO generally performs better than OvA approaches.

NEAT evolves neural networks as binary classifiers for each binary classification. An example of evolving binary classifiers (base classifiers) by NEAT is shown in the left part of Fig. 2. The voting strategy is usually used to fuse these binary classifications for multiclass classification. Each binary classifier votes to one class, and the class with the highest votes is predicted as the result. The OvO technique and base classifiers evolved by NEAT are combined for multiclass classification, i.e., OvO-NEAT. Although NEAT are effective to generate binary classifiers, the OvO-NEAT technique requires building a large number of $\binom{k}{2}$ base classifiers.

3.2.2 One-vs-All

OvA (also called One-vs-Rest or One-against-All) technique converts a k-class classification into k binary classifications. These binary classifications are constructed by using class i as the positive examples and the rest of classes j $(j = 1, ..., k, j \neq i)$ as the negative examples. Each binary classifier is used to distinguish class i from all the other $k-1$ classes. When testing an unknown example, the class with maximum prediction is considered the winner [1]. Compared to OvO, OvA provides considerable performance but requires fewer (k) classifiers.

3.2.3 Error-correcting output codes

ECOC is a class binarization method for multiclass classification, inspired by error-correcting code transmission

techniques from communications theory [21]. It encodes $\mathcal{N}$ binary classifiers to predict k classes. Each class is given an $\mathcal{N}$-length codeword according to an ECOC matrix $\mathbb{M}$. Each codeword in $\mathbb{M}$ is mapped to a certain class. An example of ECOC for $k = 4$ classes and $\mathcal{N} = 7$-bit codewords are shown in Table 1. Each column is used to train a binary classifier. When testing an unseen class, the codeword predicted by $\mathcal{N}$ classifiers is matched to the k codewords in $\mathbb{M}$. In this work, we adopt hamming distance to match predicted codeword and the ECOC codewords. The class with the minimum hamming distance is considered as the predicted class.

Unlike OvO and OvA methods that convert a multiclass classification into a fixed number of binary classifications, ECOC allows each class to be encoded with a flexible number of binary classifications, and allows extra models to act as overdetermined predictions that can result in better predictive performance [22]. The row of ECOC needs to be a unique codeword, and columns are neither identical nor complementary. In ECOC, the size of codewords (rows) is the number of classes, and thus the size of ECOC refers to the number of base classifiers in this work. The larger size ECOC provides more bits to correct errors,

Table 1 An example of ECOC for $k = 4$ classes with a size of $\mathcal{N} = 7$ bit codewords

Classes	Classifers						
	f_1	f_2	f_3	f_4	f_5	f_6	f_7
c_1	1	1	1	1	1	1	1
c_2	0	0	0	0	1	1	1
c_3	0	0	1	1	0	0	0
c_4	0	1	0	1	0	1	0

but too many classifiers cause redundancy which costs a lot of computation in training and classification.

For k classes, the minimum size of ECOC is $\lceil log_2 k \rceil$. For example, 10 classes require a minimum size of 4 bits ECOC that are sufficient for representing each class with a unique codeword. We call the ECOC with a minimum size of $\mathcal{N} = \lceil log_2 k \rceil$ as minimal ECOC. The maximum size of ECOC is $2^{k-1} - 1$ for k classes. The ECOC with a maximum size is generally called as exhaustive ECOC [21]. The upper and lower bounds of ECOC size can be expressed as:

$$\lceil log_2 k \rceil \le \mathcal{N} \le 2^{k-1} - 1, \ \mathcal{N} \in \mathbb{Z} \tag{1}$$

where $\mathbb{Z}$ is the positive integer set. Besides OvO, OvA, minimal ECOC, and exhaustive ECOC, the mid-length ECOC is another representative class binarization technique with intermediate-length code whose size is $\mathcal{N} = \lceil 10 log_2(k) \rceil$ [15].

The number of base classifiers varies as the number of classes increases using these class binarization is shown in Fig. 3. OvO requires a polynomial number of base classifiers ($O(k^2)$). However, OvA needs a linear number of classifiers ($O(k)$). For minimal ECOC and mid-length ECOC, $O(\log(k))$ binary classifiers are required. The number of base classifiers used in exhaustive ECOC is exponential ($O(2^k)$).

Algorithm 1: ECOC-NEAT for multiclass classification

Data: Construct an ECOC $\mathbb{M}$ with $\mathcal{N}$ columns and the corresponding positive dataset $\mathcal{S}_j$ and negative dataset $\bar{\mathcal{S}}_j$ for each base classifier, where $j \in [1, \mathcal{N}]$; Test dataset $\mathcal{X}$; Initialize binary classifier set $\mathbb{F}(f_1, f_2, ..., f_{\mathcal{N}})$.

1 **Output** $\mathcal{N}$ binary classifiers; predictions $\mathcal{Y}$ for each test sample x in $\mathcal{X}$.

  // Generating base classifiers by NEAT

2 **foreach** $j \in [1, \mathcal{N}]$ **do**

3   **while** $i <= \mathcal{G}/\mathcal{N}$, $\mathcal{G}$ is total generations. **do**

4   binary-NEAT evolves neural networks $\mathbb{N}_i$ for predicting the data from $\mathcal{S}_j$ and $\bar{\mathcal{S}}_j$.

5    $f_j = \text{argmax}(\mathbb{N}_i)$.

6   Update $\mathbb{F}(f_1, f_2, ..., f_{\mathcal{N}})$.

  // multiclass classification

7 **foreach** $x \in \mathcal{X}$ **do**

   // binary classification of $\mathcal{N}$ base classifiers on each test sample x.

8    $\mathbb{F}(x) \leftarrow \{f_1(x), f_2(x), \cdots, f_{\mathcal{N}}(x)\}$;

   // multiclass classification by hamming distance.

9    $\mathcal{Y} \leftarrow \text{argmin}_r \Delta(\mathbb{M}_r, \mathbb{F}(x)), r \in [1, k]$;

The exhaustive ECOC is not generally applied to the multiclass classifications with a large number of classes because it requires too many binary classifiers. The mid-length ECOC can be constructed by choosing codewords from exhaustive ECOC to satisfy row and column separation conditions. $\mathcal{N}$ columns are randomly chosen from an exhaustive code to construct the random code matrix when
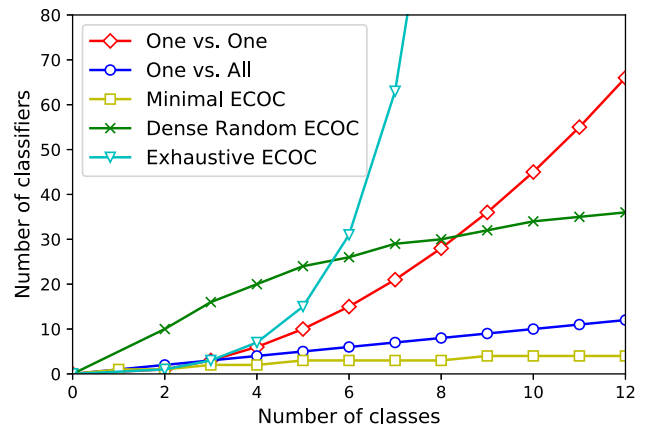

Fig. 3 Number of classifiers over the number of classes for class binarization techniques

the number of binary classifiers is $\mathcal{N}$. For example, if $k = 4, \mathcal{N} = 3$, we can choose f_1, f_2, and f_3 from the exhaustive ECOC (Table 1) to construct a mid-length ECOC. By contrast, we cannot choose f_5, f_6, and f_7 because in that case the codeword of c_1 will be exactly the same as the codeword of c_2, in which the class c_1 and c_2 can not be classified.

In general, optimized ECOC performs better than normal ECOC [23] at the same size. In this work, we investigate whether optimized minimal-ECOC outperforms minimal ECOC (see Sect. 5.2). NEAT evolves neural networks to constitute a set of binary classifiers. Hamming distance is used to determine the final prediction. The pseudo-code of ECOC-NEAT is shown in Algorithm 1.

4 Experiments

In this section, we introduce the datasets, hyperparameter configurations, implementation, and the measurements.

4.1 Datasets

In this work, we choose the three well-known datasets of *Digit* from the ski-learn package [24], *Satellite* and *Ecoli* from the machine learning repository of the University of California, Irvine (UCI) [25]. These three datasets with high quality data are prevalent and widely used in multiclass classification tasks. The properties of these three datasets are summarized in Table 2.

4.2 Experimental setup

This work compares the newly proposed ECOC-NEAT with the standard NEAT, OvO-NEAT, OvA-NEAT, and ECOC-NEAT. A hyper-parameter configuration of NEAT

Table 2 The properties of three popular datasets of *Digit*, *Satellite* and *Ecoli*

Dataset	Training samples	Test samples	classes (k)	Dimensions of feature
Digit	1617	180	10	64
Satellite	4435	2000	6	36
Ecoli	336	10-fold	8	7

Table 3 The parameter configurations of NEAT

Parameters	Value	Parameters	Value
pop_size	200	weight_mutate_rate	0.8
elitism	2	activation_mutate_rate	0.3
initial_connection	0.1	conn_delete_prob	0.1
conn_add_prob	0.8	node_delete_prob	0.1
node_add_prob	0.7	bias_mutate_rate	0.7
survival_threshold	0.2	max_fitness_threshold	1.0
max_stagnation	15	compatibility_threshold	2.5
elite_species	3	compatibility_weight_coefficient	0.6
feed_forward	True	compatibility_disjoint_coefficient	1.0

is summarized in Table 3 which are the same for evolving binary classifiers on the three datasets. The dimensions of the input layer for evolved binary classifiers equal the dimensions of feature for a dataset (the last column in Table 2). The dimension of outputs in NEAT is set to 2 for evolving binary classifiers. In the standard NEAT, the dimension of outputs equals the number of classes k for multiclass classification.

We set the number of generations as $\mathcal{G} = 3000$ for each evolution process of the standard NEAT. For a fair comparison, we apply the same total number of generations ($\mathcal{G} = 3000$) to evolve binary classifiers for these class binarization techniques. Specifically, each base classifier is generated by an evolution of ($\mathcal{G}/\mathcal{N}$) generations in NEAT if there are $\mathcal{N}$ classifiers for a class binarization technique.

We implement the standard and binary NEAT based on an open-source NEAT-Python[1]. The experiments are run on the computer with a dual 8-core 2.4 GHz CPU (Intel Haswell E5-2630-v3) and 64 GB memory.

5 Results

We show the results from the following four aspects: multiclass classification degradation, breadth evaluation, evolution efficiency, and robustness.

5.1 Multiclass classification degradation

The accuracy of multiclass classification generally decreases as the number of classes increases due to the task that becomes more difficult. We test the multiclass classification degradation of NEAT (including the standard NEAT and NEAT with class binarization) on the *Digit* dataset, in which the number of classes varies from two to ten. For example, the two-class and three-class classification predicts the digit "0, 1" and "0, 1, 2", respectively.

5.1.1 Multiclass classification degradation of the standard NEAT

The standard NEAT is used to evolve neural networks for the classification from two classes to ten classes. The experiments are repeated ten times on the *Digit* dataset. The convergence processes of the standard NEAT are
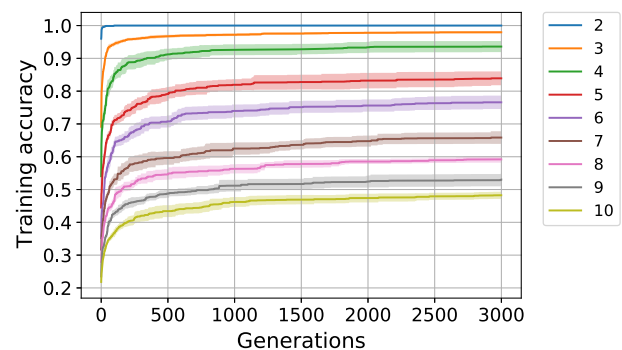

Fig. 4 The convergence processes of NEAT for the multiclass classification from two to ten classes. The shadows show 95% confidence intervals

shown in Fig. 4 where we presents the training accuracy over generations during the evolution of neural networks with 2–10 classes.

The results clearly show that the accuracy decreases dramatically as the number of classes increases. The classification of two and three classes quickly converges to the high accuracy of more than 95% with narrow confidence intervals which means their evolution processes are steady. However, the accuracy converges to the catastrophic value for the classifications with many classes. In particular, the 10-classes classification (yellow line) converges to an accuracy of less than 50% slowly. In summary, the results show that NEAT performs well for the classification with a few classes (particularly binary classification), but its performance significantly degrades over the number of classes increases.

5.1.2 Multiclass classification degradation of NEAT with class binarization

We investigate the degradation of the standard NEAT, OvO-NEAT, OvA-NEAT, and three different sizes of ECOC-NEAT (including minimal ECOC-NEAT, mid-length NEAT, exhaustive ECOC-NEAT) for multiclass classification. Figure 5 presents the performance of the standard NEAT, OvO-NEAT, OvO-NEAT, and three ECOC-NEAT for multiclass classifications with a varying number of classes from three to ten.

The results show that not only the resulting accuracy of the standard NEAT decreases dramatically but also that of NEAT with class binarization techniques decreases as the number of classifications increases. Importantly, the methods of NEAT with class binarization techniques perform slighter decreases than the standard NEAT. In particular, exhaustive ECOC-NEAT, OVO-NEAT, and mid-length ECOC-NEAT perform remarkable robustness over the number of classes increases. Moreover, they exhibit
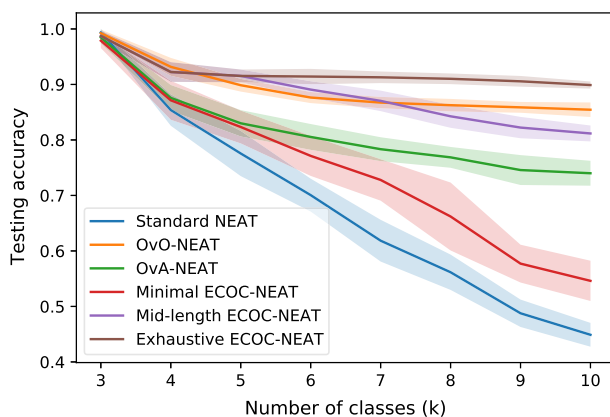

Fig. 5 Testing accuracy over number of classes for the multiclass classification methods

higher accuracy and less variance than the standard NEAT. The mid-length ECOC-NEAT with a moderate number of base classifiers provides competitive performance compared to OvO-NEAT and the exhaustive ECOC-NEAT that requires a large number of base classifiers. The exhaustive ECOC-NEAT outperforms the mid-length ECOC-NEAT that outperforms minimal ECOC-NEAT. We summarize that ECOC-NEAT methods with large size ECOC (i.e., a large number of base classifiers) generally tends to perform better than small size ECOC. Intriguingly, minimal ECOC-NEAT with a few bases learners still significantly performs better than the standard NEAT for multiclass classification.

5.2 Comprehensive comparison

We investigate the standard NEAT, OvO-NEAT and OvA-NEAT and the proposed ECOC-NEAT methods with different codes including the minimal, mid-length and exhaustive code on the three datasets. Specially, we apply the mid-lengths ECOC-NEAT with different sizes to investigate the relationship between the size of ECOC-NEAT and their resulting accuracy. The performance of these methods is shown in Table 4 where we presents (1) testing accuracy (accuracy on test set), (2) variance of testing accuracy over ten repetitions, (3) training accuracy on the training set, (4) average training accuracy of each base classifier, and (5) average training time per generation.

The results show that NEAT with class binarization techniques significantly outperform the standard NEAT in terms of accuracy. ECOC-NEAT even the minimal ECOC-NEAT performs higher accuracy than the standard NEAT on the three datasets. The exhaustive ECOC-NEAT with the largest number of base classifiers performs the smallest variances that represent the strong robustness. Conversely, the minimal ECOC-NEAT with a few binary classifiers performs large variances that mean the fluctuating performance.

The average training accuracy of each base classifier shows the performance of each evolved binary classifier on the training dataset. The binary classifiers in OvO-NEAT perform the best average training accuracy because it decomposes multiclass classifications into simple binary classification tasks. The evolved binary classifiers in ECOC-NEAT methods perform lower average accuracy than OvO-NEAT and OvA-NEAT because the binary classifications in ECOC-NEAT are generally challenging and each classifier in ECOC-NEAT is assigned a few $(\mathcal{G}/\mathcal{N})$ generations to evolve. However, the ECOC-NEAT methods still perform high accuracy for multiclass classification due to the high quality of ensemble in ECOC.

Table 4 Comparison of different methods on the three datasets of *Digit* (10 classes), *Satellite* (6 classes), and *Ecoli.* (8 classes)

Dataset	Method		Number of classifiers	Testing accuracy	Variance	Training accuracy	$\overline{\mathcal{A}_b}$	Average training time/ Generation(s)
Digit	Existing	Standard NEAT	1	0.449	9.56×10^{-4}	0.484	0.484	13.74
		OvO-NEAT	45	<u>0.866</u>	4.94×10^{-4}	**0.953**	**0.989**	**0.99**
		OvA-NEAT	10	<u>0.740</u>	10.46×10^{-4}	0.820	0.976	8.40
	Ours	Minimal ECOC-NEAT	4	<u>0.535</u>	72.01×10^{-4}	0.614	0.865	10.49
		10-bit ECOC-NEAT	10	<u>0.651</u>	15.78×10^{-4}	0.724	0.860	8.49
		45-bit ECOC-NEAT	45	<u>0.819</u>	9.04×10^{-4}	0.876	0.837	5.62
		100-bit ECOC-NEAT	100	<u>0.845</u>	6.77×10^{-4}	0.894	0.812	4.96
		250-bit ECOC-NEAT	250	<u>0.876</u>	2.76×10^{-4}	0.908	0.793	4.60
		Exhaustive ECOC-NEAT	511	**<u>0.899</u>**	**0.95** $\times10^{-4}$	0.909	0.783	4.53
Satellite	Existing	Standard NEAT	1	0.754	0.99×10^{-4}	0.774	0.774	5.09
		OvO-NEAT	28	<u>0.842</u>	0.79×10^{-4}	**0.914**	**0.989**	**0.12**
		OvA-NEAT	8	<u>0.787</u>	1.53×10^{-4}	0.848	0.979	0.75
	Ours	Minimal ECOC-NEAT	3	<u>0.765</u>	2.28×10^{-4}	0.816	0.922	1.11
		8-bit ECOC-NEAT	8	<u>0.790</u>	3.24×10^{-4}	0.844	0.926	0.94
		15-bit ECOC-NEAT	15	<u>0.828</u>	3.90×10^{-4}	0.870	0.922	0.84
		28-bit ECOC-NEAT	28	**<u>0.849</u>**	0.79×10^{-4}	0.881	0.917	0.73
		40-bit ECOC-NEAT	40	<u>0.848</u>	**0.46** $\times10^{-4}$	0.885	0.914	0.68
		60-bit ECOC-NEAT	60	<u>0.848</u>	2.16×10^{-4}	0.885	0.910	0.62
		Exhaustive ECOC-NEAT	127	<u>0.837</u>	0.88×10^{-4}	0.873	0.900	0.55
Ecoli.	Existing	Standard NEAT	1	0.754	0.99×10^{-4}	0.774	0.774	5.09
		OvO-NEAT	28	<u>0.842</u>	0.79×10^{-4}	**0.914**	**0.989**	**0.12**
		OvA-NEAT	8	<u>0.787</u>	1.53×10^{-4}	0.848	0.979	0.75
	Ours	Minimal ECOC-NEAT	3	<u>0.765</u>	2.28×10^{-4}	0.816	0.922	1.11
		8-bit ECOC-NEAT	8	<u>0.790</u>	3.24×10^{-4}	0.844	0.926	0.94
		15-bit ECOC-NEAT	15	<u>0.828</u>	3.90×10^{-4}	0.870	0.922	0.84
		28-bit ECOC-NEAT	28	**<u>0.849</u>**	0.79×10^{-4}	0.881	0.917	0.73
		40-bit ECOC-NEAT	40	<u>0.848</u>	**0.46** $\times10^{-4}$	0.885	0.914	0.68
		60-bit ECOC-NEAT	60	<u>0.848</u>	2.16×10^{-4}	0.885	0.910	0.62
		Exhaustive ECOC-NEAT	127	<u>0.837</u>	0.88×10^{-4}	0.873	0.900	0.55

Each method is run ten times and token an average of results. The total generation of each method is identical $\mathcal{G}=3000$. $\mathcal{N}$-bit ECOC-NEAT represents different sizes of mid-length ECOC-NEAT that is $\mathcal{N}$ base classifiers. $\overline{\mathcal{A}_b}$ represents the average training accuracy of binary classifiers

The underlined values represent the testing accuracy of multiclass classification methods with binary classification. The bold values are the best values
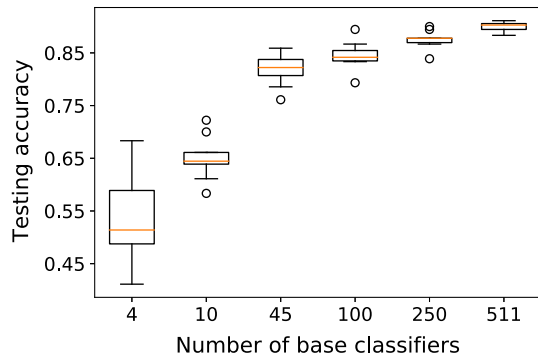
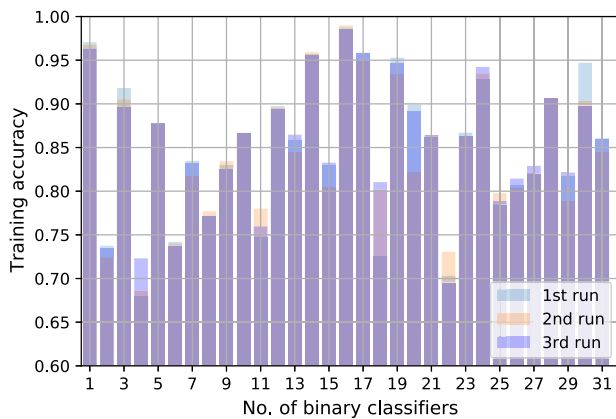**Fig. 6** Testing accuracy over ECOC-NEAT size on *Digit*

Fig. 7 Training accuracy of the 31 binary classifiers in an exhaustive ECOC-NEAT on the *Satellite* dataset for three repetitions

NEAT in these methods takes different computation times to evolve binary classifiers. The standard NEAT takes much more computation time per generation to evolve classifiers than the NEAT with class binarization techniques.

5.3 Size of ECOC-NEAT

The size of ECOC performs a significant influence on their performance for multiclass classification [26]. To further

observe the influence of the size of ECOC on their performance, we visualize the testing accuracy and variance over the size of ECOC (the results in Table 4) in Fig. 6. The visualization shows that testing accuracy increases as the number of base classifiers increases and the small size ECOC-NEAT performs fluctuating testing accuracy. A similar observation can be illustrated from the results on the *Satellite* and *Ecoli* datasets (Table 4).

5.4 Quality of ECOC-NEAT

Besides the size of ECOC, the quality of ECOC is another crucial factor for the performance of ECOC-NEAT. The minimal ECOC-NEAT with a few base classifiers generally perform sensitive to the quality of ECOC. Thus, we concentrate on the quality of the minimal ECOC-NEAT.

5.4.1 On the *Satellite* dataset

ECOC-NEAT with high training accuracy binary classifiers generally performs high testing accuracy for multiclass classification. The binary classification tasks in an ECOC-NEAT are generally with various difficulty. The exhaustive ECOC for the *Satellite* dataset with $k = 6$ classes is with 31 columns (see Table 4). We run an exhaustive ECOC-NEAT to evolve the 31 binary classifiers on the *Satellite* dataset for three repetitions. The training accuracy of these 31 binary classifiers is shown in the bar chart of Fig. 7. The results show that these binary classifiers in the exhaustive ECOC-NEAT perform significant different accuracy from around 70% to 98%.

For the *Satellite* dataset with $k = 6$ classes, the minimal ECOC-NEAT needs a minimum of 3-bit codeword (three columns) to construct the ECOC. We random choose three columns from the 31 columns of the exhaustive ECOC to construct minimal ECOCs. For an exhaustive ECOC with 31 columns, there are $\binom{31}{3} = 4495$ combinations, and 420 out of these 4495 combinations are available minimal ECOCs that satisfy both row and column conditions. We
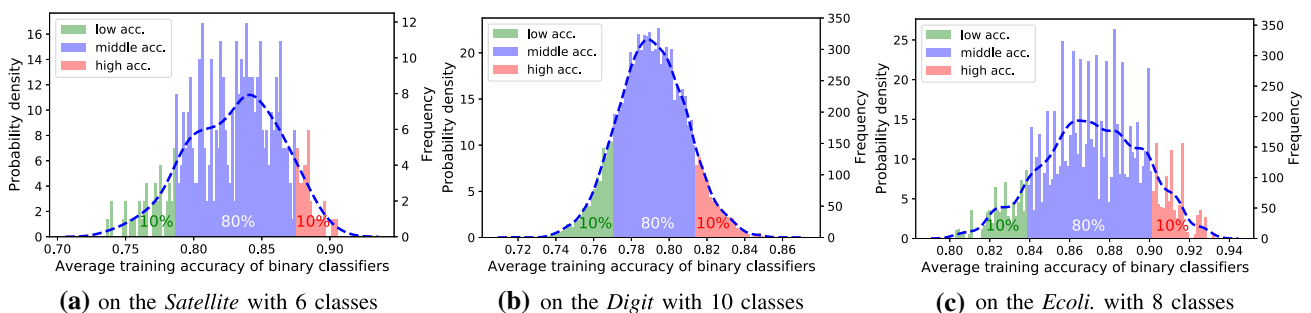

(a) on the *Satellite* with 6 classes   **(b)** on the *Digit* with 10 classes   **(c)** on the *Ecoli.* with 8 classes

Fig. 8 Distribution of all minimal ECOC-NEAT in terms of average classifiers training accuracy on the three datasets of *Satellite*, *Digit*, *Ecoli.*. The frequency of the right vertical axis represents the number of ECOC

Table 5 The performance of minimal ECOC-NEAT of different qualities on the three datasets of *Satellite*, *Digit*, and *Ecoli*. $\overline{\mathcal{A}_b}$ represents the average training accuracy of binary classifiers

Dataset	Minimal ECOC	Testing accuracy	Variance	Trainin accuracy	$\overline{\mathcal{A}_b}$
Satellite	Middle-	0.704	9.79×10^{-4}	0.714	0.881
	High-	**0.774**	**3.55 $\times 10^{-4}$**	**0.793**	**0.915**
	Low-	0.632	33.77×10^{-4}	0.653	0.848
Digit	Middle-	0.535	72.01×10^{-4}	0.614	0.865
	High-	**0.647**	31.44×10^{-4}	**0.733**	**0.912**
	Low-	0.483	**6.83 $\times 10^{-4}$**	0.538	0.814
Ecoli.	Middle-	0.765	2.28×10^{-4}	0.816	0.922
	High-	**0.818**	1.86×10^{-4}	**0.867**	**0.953**
	Low-	0.678	**1.76 $\times 10^{-4}$**	0.740	0.870

The bold and underlined values are the best values

(a) Satellite (6 classes)   **(b)** Digit (10 classes)   **(c)** Ecoli. (8 classes)
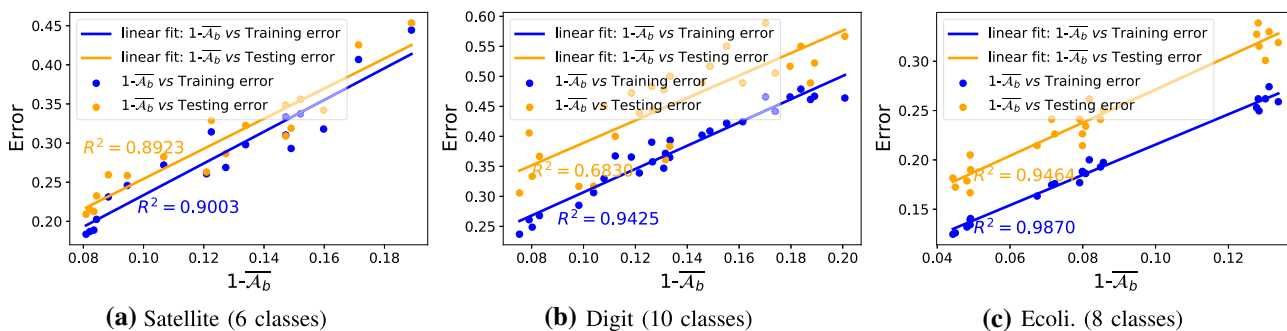
Fig. 9 Training/Testing error and average training error of binary classifiers on the *Satellite* problem. Distribution of all minimal ECOC-NEAT in terms of average classifiers training accuracy on the three datasets of *Satellite*, *Digit*, *Ecoli.*. The frequency of the right vertical axis represents the number of ECOC. The lines are applied to fit the data, and R^2 is the goodness of fit

run all 420 minimal ECOC-NEAT on the *Digit* dataset. Fig. 8a shows the distribution of average training accuracy of binary classifiers (noted as $\overline{\mathcal{A}_b}$) in these 420 minimal ECOCs. These 420 minimal ECOCs perform different qualities in terms of their average training accuracy of binary classifier from around 70% to 90%. We divide these 420 minimal ECOCs into the three-level performance of low, middle, high accuracy with the ratio of 10%, 80%, and 10%, respectively. These 10% minimal ECOCs with high accuracy are the optimized minimal ECOCs. The results indicate that different ECOCs perform significantly accuracy and the quality of ECOC is crucial for the high accuracy of binary classifiers.

Moreover, we randomly choose minimal ECOCs with low, middle, and high accuracy, respectively. Each minimal ECOC-NEAT evolves three binary classifiers (three columns in each minimal ECOC) with an evolution of total 3000 generations for multiclass classification, which results is shown in Table 5. The results indicate that the average training accuracy of binary classifiers significantly impacts the testing accuracy. The optimized minimal ECOC-NEAT performs a testing accuracy of 0.7735 that is much higher

than the low accuracy minimal ECOC-NEAT and the standard NEAT (0.6377 in Table 4) for 6-classes classification on the *Satellite* dataset. Conversely, the low accuracy minimal ECOC-NEAT perform a similar testing accuracy with the standard NEAT.

Finally, we randomly choose 6 ECOCs from high, middle, low accuracy ECOCs, respectively, that is 18 various ECOCs in total, to observe the relationship between their training/testing error and average training error of binary classifiers ($1-\overline{\mathcal{A}_b}$), as shown in Fig. 9a. The lines are applied to fit the data, and indicate that the training/testing error is linear with the average training error of binary classifiers. The optimized minimal ECOCs perform the results that are shown in the left-bottom points with low training/testing error and low average training error of binary classifiers.

5.4.2 On the *Digit* dataset

For the *Digit* dataset with 10 classes, an exhaustive ECOC and a minimal ECOC consists of 511 base classifiers and four base classifiers, respectively (as shown in Table 4). An

exhaustive ECOC with 511 columns can be used to construct a large number of $\binom{511}{4} = 2,807,768,705$ 4-bit possible minimal ECOCs (4 columns) that is a huge amount of work and not necessary to be investigated. In this work, we random choose 10,000 minimal ECOCs to investigate the performance of various minimal ECOCs on the *Digit* dataset. The distribution of the average training accuracy of binary classifiers is shown in Fig. 8b. Interestingly, the distribution looks like a normal distribution. We divide these minimal ECOCs into the three-level performance of low, middle, high accuracy with the ratio of 10%, 80%, and 10%, respectively. As the standard NEAT with an evolution of 3000 generations, each classifier of these 511 binary classifiers is generated by an evolution of $\lceil 3000/511 \rceil \approx 6$ generations. Theoretically and empirically, the average training accuracy of binary classifiers can be improved with a longer evolution than 6 generations, and thus lead to the higher accuracy for multiclass classification on the *Digit* dataset.

We randomly choose minimal ECOCs from low, middle, high accuracy (in Fig. 8b), respectively. These minimal ECOC-NEAT evolve binary classifiers with an evolution of $3000/4 = 750$ generations. The results of these minimal ECOC-NEAT on the *Digit* dataset is shown in Table 5. The high accuracy 4-bit ECOC-NEAT performs a remarkable testing accuracy that is comparable with the 10-bit mid-length ECOC-NEAT (a testing accuracy of 0.6506, see Table 4), and saves 60% classifiers (from 10 to 4). The low accuracy ECOC-NEAT still perform a low testing accuracy of 0.4832 that is only a little superior to the standard NEAT.

We randomly choose 9 minimal ECOCs from low, middle, high accuracy, respectively, that is 27 various ECOCs in total, to investigate the relationship between their training/testing error and average training error of binary classifiers $(1 - \overline{\mathcal{A}_b})$, as shown in Fig. 9b. The lines are applied to fit the data, and indicate that the training/testing error is linear with the average training error of binary classifiers. The 27 minimal ECOC-NEAT generate binary classifiers by an evolution of $3000/4 = 750$ generations and thus the binary classifiers performs higher average training accuracy (1 - average training error of binary classifiers) than the results in Fig. 8b.

a) on the Ecoli. Dataset For the *Ecoli.* dataset with 8 classes, an exhaustive ECOC-NEAT and a minimal ECOC-NEAT consists of 127 and 3 base classifiers (3-bit), respectively. An exhaustive ECOC can be used to construct a large number $(\binom{127}{3} = 333,375)$ of minimal ECOCs. In this work, we randomly choose 10,000 minimal ECOCs. The distribution of average training accuracy of
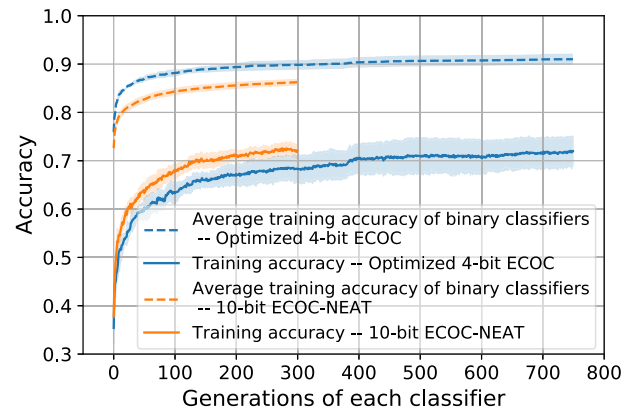

Fig. 10 The training accuracy and average training accuracy of binary classifiers of 4-bit optimized minimal ECOC-NEAT and 10-bit mid-length ECOC-NEAT on the *Digit* dataset. The lines and shadow represent the mean and 95% confidence intervals for 10 repetitions

binary classifiers is shown in Fig. 8c. We categorize these minimal ECOC-NEAT into three levels of high, middle, low average training accuracy of binary classifiers.

Moreover, we randomly choose a minimal ECOC from low, middle, high accuracy, respectively, and run the minimal ECOC-NEAT to evolve binary classifiers with an evolution of 1000 (3000/3) generations. The results of the low, middle, high accuracy (optimized) minimal ECOC-NEAT on the *Ecoli* dataset are shown in Table 5. The high accuracy 3-bit minimal ECOC-NEAT performs a test accuracy near with 15-bit mid-length ECOC-NEAT. The low accuracy ECOC-NEAT performs a low test accuracy of 0.6782 that is even lower than that of the standard NEAT.

In addition, we randomly choose 7 minimal ECOCs from low, middle, high accuracy (optimized) minimal ECOCs (i.e., 21 various ECOCs in total) to validate the relationship between the quality of ECOCs and their training/testing error, as shown in Fig. 9c. The lines that fit the results and indicate the linear relation between the quality of ECOCs and their training/testing error.

To summarize, we conclude that a high quality ECOC generally performs high testing accuracy. It is crucial to design a high quality ECOC for multiclass classification of neuroevolution approaches.

5.5 Evolutionary efficiency

We observe the convergence of training accuracy and average training accuracy of binary classifiers during the evolution. We randomly choose an optimized minimal ECOC-NEAT from Table 5 and a 10-bit ECOC-NEAT from Table 4, and run them 10 repetitions on the *Digit* dataset. The minimal ECOC-NEAT and 10-bit mid-length ECOC-NEAT generate binary classifiers with an evolution
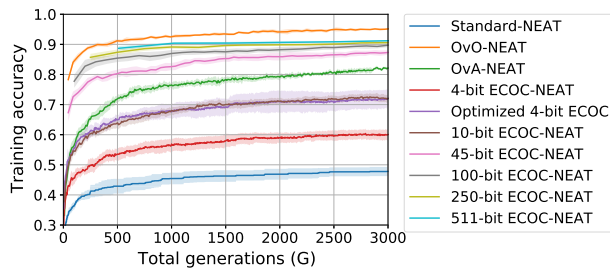
Fig. 11 Training accuracy of the standard NEAT and the NEAT with class binarization techniques over generations on the *Digit* dataset for 10 classes classification

of 750 generations and 300 generations, respectively. The results are shown in Fig. 10.

The results show that the training accuracy performs a significant similar convergence process with the average training accuracy of binary classifiers. Both of them dramatically increase in the beginning and gradually converge to a stable value over generations. The high accuracy 4-bit minimal ECOC-NEAT performs a training accuracy of 72% approximately, which performs even higher accuracy than the 10-bit mid-length ECOC-NEAT with a training accuracy of 71%.

Moreover, we compare the training accuracy of the standard NEAT and NEAT with class binarization techniques during the evolution, as shown in Fig. 11. The number of generations for each evolution of ECOC-NEAT is $\mathcal{G}/\mathcal{N}$ which is different for various ECOC-NEAT. To compare various ECOC-NEAT in the same scale, we apply proportional scaling to match an identical x-axis. For example, 10-bit mid-length ECOC-NEAT with the evolution of 300 generations for each binary classifier in Fig. 10 is scaled 10 times in Fig. 11.

The results show that NEAT with class binarization techniques perform significantly better in terms of accuracy than the standard NEAT for multiclass classification. OvO-NEAT, exhaustive ECOC-NEAT, mid-length ECOC-NEAT (including 250-bit, 100-bit, 45-bit ECOC-NEAT) perform remarkable training accuracy. The NEAT with large size ECOC (e.g., exhaustive ECOC-NEAT, OvO-NEAT) generally performs better than the NEAT with small size ECOC (e.g., 4-bit ECOC-NEAT). Compared to the normal 4-bit ECOC-NEAT with a training accuracy of 60% approximately, the optimized 4-bit ECOC-NEAT perform an efficient multiclass classification with a training accuracy of 72% approximately. Moreover, the optimized 4-bit ECOC-NEAT performs significantly similar evolution process (the purple line) with the 10-bit ECOC-NEAT (the brown line). The results demonstrate that the size and quality of ECOC are crucial for the multiclass classification performance of ECOC-NEAT.
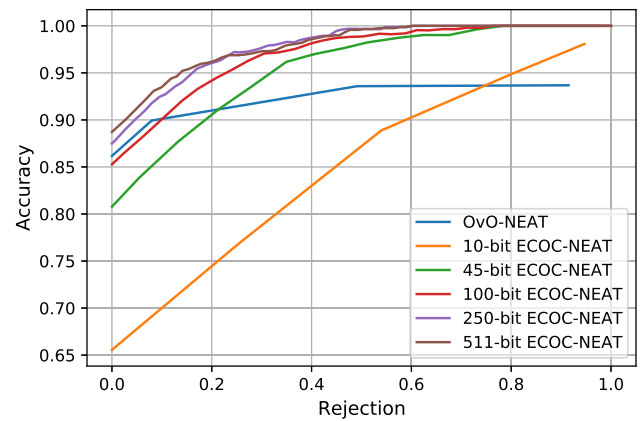

Fig. 12 Accuracy rejection curves of OvO-NEAT and other ECOC-NEAT on the *Digit* dataset

5.6 Robustness

Robustness is an important measurement for the evaluation of multiclass classification. The ECOC-NEAT usually performs a remarkable ability to correct errors for multiclass classification. Gunjan Verma and Ananthram Swami applied ECOC to improve the adversarial robustness of deep neural networks [27]. Although OvO-NEAT performs outstanding for multiclass classification, the robustness of OvO-NEAT against errors is insufficient compared to ECOC-NEAT. In this work, we apply the measure of Accuracy-Rejection curve to analyse the robustness of the NEAT with class binarization techniques. Figure 12 shows the accuracy-rejection curve of OvO-NEAT and other ECOC-NEAT.

The large size ECOCs perform better than the small size ECOCs no matter whether the rejection rates are low or high. Large size ECOC-NEAT always outperforms OvO-NEAT, which means they have consistently stronger robustness against errors than OvO-NEAT. Comparing the small size of 10-bit ECOC-NEAT with OvO-NEAT, there is an intersection between two lines. The lines of the large size ECOC intersect the line of OvO-NEAT at the small values of rejections. From a rejection rate of the intersection onwards, ECOC-NEAT outperforms OvO-NEAT. For example, at rejection rates greater than 80%, even 10-bit ECOC-NEAT outperforms OvO-NEAT, which means 10-bit ECOC-NEAT gives 20% of the test samples pretty convincing predictions (with testing accuracy of 95%). Briefly, ECOC-NEAT has strong robustness against errors, especially with long codes. By contrast, the robustness of OvO-NEAT seems weak.

ECOC-NEAT performs strong robustness that its base classifiers are complement each other when the number of base classifiers decreases. In this work, we investigate the robustness of the performance of ECOC-NEAT and OvO-NEAT when their number of base classifiers decreases. The

results are shown in Fig. 13, where the size of ECOC and OvO decrease from 45-bit (45 base classifiers) to 1-bit (one classifier). We randomly choose base classifiers from 45-bit ECOC-NEAT and OvO-NEAT to construct various size ECOC-NEAT and OvO-NEAT with ten repetitions. The results show that the testing accuracy of OvO-NEAT declines almost linearly as the number of base classifiers decreases. However, the accuracy of ECOC-NEAT decreases slightly as the number of base classifiers decreases. In particular, the accuracy of ECOC-NEAT hardly decreases when ECOC-NEAT is with a little fewer base classifiers, e.g., 40-bit ECOC-NEAT. The ECOC-NEAT with 22 base classifiers, that is half of 45 base classifiers, still obtains a testing accuracy of approximately 70% that dropping by 12% from the testing accuracy of 45-bit ECOC-NEAT (82%). However, OvO-NEAT with 22 base classifiers performs 45% testing accuracy that dropping by 41% from the testing accuracy of 45-bit OvO-NEAT (86%). This finding illustrates that ECOC-NEAT performs better robustness than OvO-NEAT when they ensemble fewer base classifiers for multiclass classification.

OvO is a decent class binarization technique for multiclass classification with high accuracy, low variance, and efficient training process [3, 6], but requires too many classifiers ($O(k^2)$). The large size ECOC usually performs high accuracy, low variance, and strong robustness [28, 29]. An optimized minimal ECOC significantly outperforms a normal constructed ECOC [23].

In summary, we recommend OvO-NEAT and ECOC-NEAT with a great number of binary classifiers (e.g. mid-length ECOC-NEAT, or exhaustive ECOC-NEAT with moderate classes) for the tasks when a considerable number of generations is allowed. For the tasks that only limited generations are allowed, we recommend optimized ECOC-NEAT with a small number of binary classifiers.

6 Discussions and future work

6.1 Discussions

In this section, we analyse the classification performance of these methods on different classes and the network complexity of base classifiers.

6.1.1 Behavior analysis

We observe the classification performance on each class of these methods by analyzing the results of the standard NEAT and the NEAT with class binarization techniques on the *Digit* dataset[2]. We apply the widely used metrics of
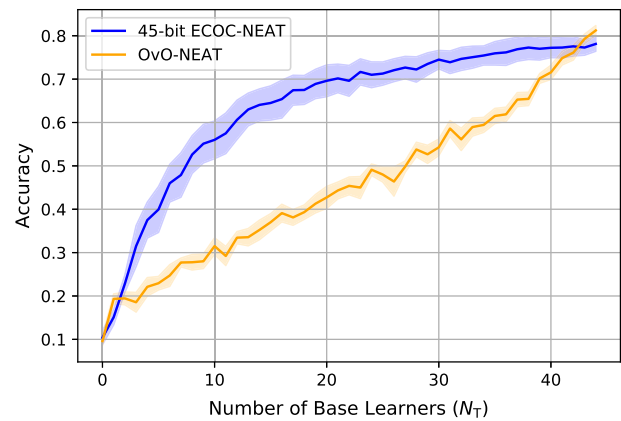

Fig. 13 Testing accuracy of ECOC-NEAT and OvO-NEAT with various number of base classifiers. The experiments are repeated 10 times and take an average of the testing accuracy. The shadow represents 95% confidence intervals
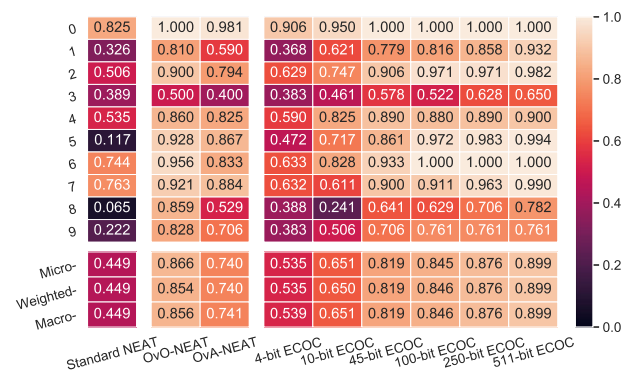

Fig. 14 Precision heatmap of these methods on the *Digit* dataset. Rows from 0 to 9 are precision on the digit class from "0" to "9". Rows 10, 11, 12 present micro-averaging precision, weighted-averaging precision, and macro-averaging precision, respectively. Columns represent various methods

precision, recall, and F1-score to evaluate the classification on each class of these methods. Moreover, we adopt popular averaging methods for precision, recall, and F1-score, resulting in a set of different average scores (macro-averaging, weighted-averaging, micro-averaging), see more details of these averaging methods in [30]. We conduct experiments for ten repetitions and take an average of the results. The heatmaps of the precision, recall, and F1-score of these methods are visualized in Figs. 14, 15, 16, respectively.

The classification precision on each class of the *Digit* dataset from "0" to "9" is shown in the heatmap of Fig. 14. The results show that the difficulty of classifications on different digits is diverse. Specifically, the digit "0" is predicted by all these methods with high accuracy of more than 90%. All these methods perform low testing

---

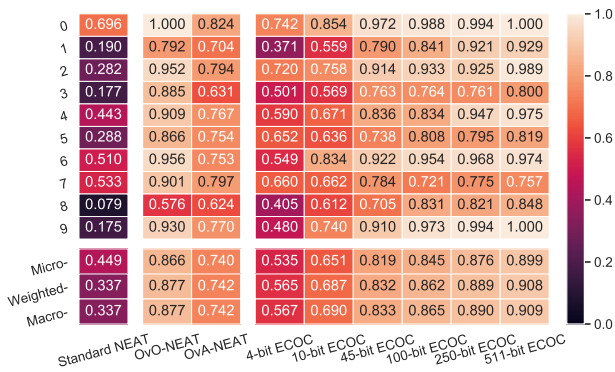[2] It does not need to analyze the results on all three datasets.

Fig. 15 Recall heatmap of different methods on the *Digit* dataset. Rows 0 to 9 present the recall of digit class "0" to "9". Rows 10 to 12 present micro-averaging recall, weighted-averaging recall, and macro-averaging recall, respectively. Columns represent different methods
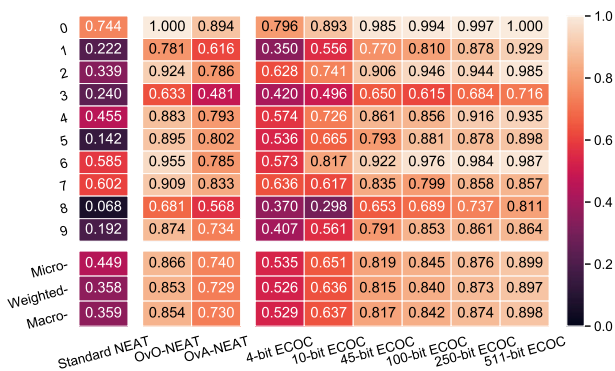

Fig. 16 F1-score Heatmap of different multiclass classification methods. Rows 0 to 9 present the F1-score of digit class from "0" to "9". Rows 10 to 12 present micro-averaging, weighted-averaging, and macro-averaging F1-score, respectively. Columns represent different methods

accuracy on the digit "3" and "8". The other digits are classified with diverse accuracies that are basically desired. The larger size ECOC-NEAT generally performs higher precision than the small size ECOC-NEAT. For example, a micro-averaging precision of 0.5350 for 4-bit ECOC-NEAT increases to 0.8189 for 45-bit ECOC-NEAT. All ECOC-NEAT including the small size 4-bit ECOC-NEAT outperform the standard NEAT. The precision of the standard NEAT once again verifies its low performance for multiclass classification. Exceptionally, the standard NEAT predict the digit "0" with a decent accuracy, which verifies that the digit "0" is distinctly predicted.

Figure 15 shows the recall heatmap of different methods for classifying the digit class "0" to "9". The recall heatmap shows consistent results with the precision heatmap. For example, the recall of digit classes "3" and "8" are usually the low for all these methods.
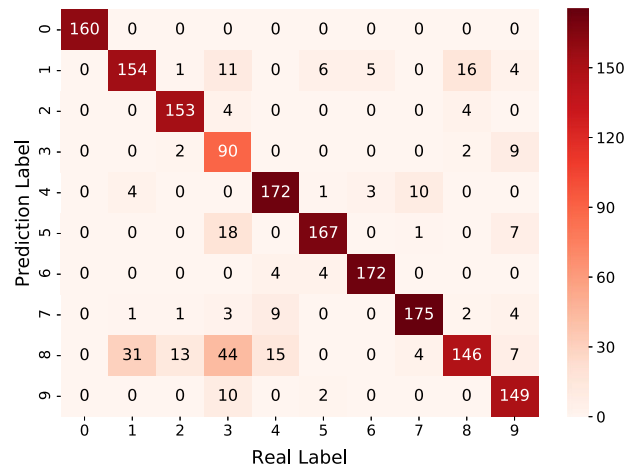

Fig. 17 The heatmap of predicted label by OvO-NEAT and real label on the *Digit* dataset

F1-score is the harmonic mean of precision and recall to evaluate model performance comprehensively, which conveys a balance between precision and recall. The F1-score of different methods on the *Digit* dataset is shown in Fig. 16. The recall heatmap shows consistent results with the precision and recall heatmaps.

It is worth noticing that OvO-NEAT performs a high precision on the digit "8" but a low precision on the digit "3" in Fig. 14. By contrast, its recall on the digit "8" is lower compared to the digit "3" in Fig. 15. We suppose that there are recognition errors between these two categories, and therefore observe the predicted label of OvO-NEAT and real label to verify this hypothesis, as shown in Fig. 17.

The results show that OvO-NEAT often incorrectly predicts the digit "3" as "8". Specifically, 44 digits of "3" are incorrectly predicted as the digit "8". This explains that these methods perform low testing accuracy on the digit "3" and "8". Intuitively, the digit "3" and "8" have similar shapes, and they are even incorrectly recognized by human.

In summary, the three heatmaps of precision, recall, and F1-score reveal consistent conclusions that 1) NEAT with class binarization techniques, particularly ECOC-NEAT and OvO-NEAT, outperform the standard NEAT for multiclass classification, 2) the large ECOC-NEAT generally performs high precision, recall, and F1-score, 3) NEAT (including the standard NEAT, OvO-NEAT, ECOC-NEAT) techniques perform diverse on different classes and large size ECOC-NEAT perform robust for the classification with different classes.

Table 6 Network complexity of generated classifiers by different NEAT-based methods for different number of classes on the *Digit* dataset. The value and the the value in the bracket are the average

total number of all base classifiers and the average number of each base classifier over ten repetitions, respectively

		Number of Classes							
		3	4	5	6	7	8	9	10
Standard NEAT	# Classifiers	1	1	1	1	1	1	1	1
	Generations	3000(3000)	3000(3000)	3000(3000)	3000(3000)	3000(3000)	3000(3000)	3000(3000)	3000(3000)
	Nodes	43(43)	68(68)	61(61)	55(55)	60(60)	63(63)	68(68)	72(72)
	Connections	150(150)	308(308)	211(211)	172(172)	133(133)	121(121)	132(132)	177(177)
OvO-NEAT	# Classifiers	3	6	10	15	21	28	36	45
	Generations	3000(1000)	3000(500)	3000(300)	3000(200)	3003(143)	2996(107)	2988(83)	3015(67)
	Nodes	81(27)	159(27)	249(25)	396(26)	506(24)	657(23)	857(24)	1039(23)
	Connections	205(68)	282(47)	505(51)	658(44)	744(35)	916(33)	1177(33)	1387(31)
OvA-NEAT	# Classifiers	3	4	5	6	7	8	9	10
	Generations	3000(1000)	3000(750)	3000(600)	3000(500)	3003(429)	3000(375)	2997(333)	3000(300)
	Nodes	109(36)	144(36)	186(37)	221(37)	260(37)	303(38)	325(36)	355(36)
	Connections	341(114)	464(116)	661(132)	649(108)	598(85)	967(121)	836(93)	931(93)
Minimal ECOC-NEAT	# Classifiers	2	2	3	3	3	3	4	4
	Generations	3000(1500)	3000(1500)	3000(1000)	3000(1000)	3000(1000)	3000(1000)	3000(750)	3000(750)
	Nodes	77(39)	98(49)	132(44)	142(47)	139(46)	120(40)	166(42)	174(44)
	Connections	207(104)	492(246)	475(158)	479(160)	471(157)	493(164)	518(130)	540(135)
Mid-length ECOC-NEAT	# Classifiers	3	7	15	26	29	30	32	34
	Generations	3000(1000)	3003(429)	3000(200)	2990(115)	2987(103)	3000(100)	3008(94)	2992(88)
	Nodes	107(36)	266(38)	477(32)	739(28)	796(27)	831(28)	848(27)	881(26)
	Connections	286(95)	783(112)	1012(67)	1300(50)	1450(50)	1347(45)	1353(42)	1416(42)
Exhaustive ECOC-NEAT	# Classifiers	3	7	15	31	63	127	255	511
	Generations	3000(1000)	3003(429)	3000(200)	3007(97)	3024(48)	3048(24)	3060(12)	3066(6)
	Nodes	107(36)	266(38)	477(32)	836(27)	1446(23)	2988(24)	4740(19)	8711(17)
	Connections	286(95)	783(112)	1012(67)	1317(42)	1903(30)	3060(24)	5060(20)	8688(17)

Table 7 Network complexity of generated classifiers by different NEAT-based methods on the *Satellite* dataset

Method	Generations	#Classifiers	Nodes	Connections
Standard NEAT	3000 (3000)	1	44 (**44**)	156 (**156**)
OvO-NEAT	3000 (200)	15	325 (22)	1095 (73)
OvA-NEAT	3000 (500)	6	151 (25)	975 (163)
Minimal ECOC	3000 (1000)	3	82 (27)	566 (189)
6-bit ECOC	3000 (500)	6	145 (24)	775 (129)
10-bit ECOC	3000 (300)	10	226 (23)	831 (83)
15-bit ECOC	3000 (200)	15	303 (20)	1076 (72)
20-bit ECOC	3000 (150)	20	397 (20)	1119 (56)
Exhaustive ECOC	3007 (97)	**31**	**529** (17)	**1252** (40)

The bold values are the key values of different elements for different methods

6.1.2 Network complexity

Network complexity offers an insight into the analysis of the mechanisms of NEAT with class binarization techniques for multiclass classification. We investigate how the number of nodes and connections influence classification performance. Table 6 shows the network complexity of generated classifiers by different NEAT-based methods for a different number of classes on the *Digit* dataset. These

Table 8 Network complexity of generated classifiers by different NEAT-based methods on the *Ecoli.* dataset

Method	Generations	#Classifiers	Nodes	Connections
Standard NEAT	3000 (3000)	1	29 **(29)**	262 **(262)**
OvO-NEAT	2996 (107)	28	170 (6)	254 (9)
OvA-NEAT	3000 (375)	8	89 (11)	323 (40)
Minimal ECOC	3000 (1000)	3	44 (15)	272 (91)
8-bit ECOC	3000 (375)	8	106 (13)	457 (57)
15-bit ECOC	3025 (200)	15	183 (12)	634 (42)
28-bit ECOC	2996 (107)	28	309 (11)	880 (31)
40-bit ECOC	3000 (75)	40	407 (10)	1007 (25)
60-bit ECOC	3000 (50)	60	556 (9)	1202 (20)
Exhaustive ECOC	3048 (24)	**127**	**964** (8)	**1321** (10)

The bold numbers are the maximum values of different elements for different methods

experiments are repeated ten times. The network complexity on the *Satellite* and *Ecoli.* dataset are presented in Table 7 and 8. We observe the average total number of nodes and connections of all base classifiers over ten repetitions, and the average number of nodes and connections of each base classifier (the value in the bracket). For example, the exhaustive ECOC-NEAT generate three base classifiers with an average total number of 107 nodes and 286 connections for 3 classes, and an average number of 36 nodes and 95 connections for each base classifier over ten repetitions.

As the number of classes increases, it is reasonable to generate a complex neural network with more nodes and connections for more complicated patterns. However, the results show that the standard NEAT struggles to generate the neural networks with augmented nodes and connections as the number of classes increases, which basically causes its dramatic multiclass classification degradation. We hypothesis that the standard NEAT tend to eliminate the evolved neural networks with more nodes and connections during the evolution. In contrast, NEAT with class binarization techniques tend to generate neural networks with more nodes and connections as the number of classes increases for remarkable multiclass classification. Although ECOC-NEAT often generates the base classifiers with fewer and fewer nodes and connections as the number of classes increases, the increasing number of binary classifiers leads to the increasing total number of nodes and connections that contribute to the remarkable performance of multiclass classification. For example, the base classifier evolved by the exhaustive ECOC-NEAT for 3 classes has an average of 36 nodes and 95 connections, but that for 10 classes has an average of only 17 nodes and 17 connections. However, the total nodes and connections increase from 107 and 286 to 8711 and 8688, respectively, for 10 classes classification.

6.2 Future work

Although this work investigates the different class binarization techniques, there are still multiple open issues and possible future work that may provide new insights into ECOC-NEAT. First, the ECOC-NEAT needs to train a lot of binary classifiers, which generally takes a lot of training time. Second, the hamming distance for matching the predicted codeword and ECOC codewords is a basic matching strategy that needs to be improved. Third, the ECOC still needs to be improved with different code design. We would like to improve the performance of ECOC-NEAT from the aspects of:

- using sparse codes (i.e., $\mathbb{M} \in \{1, -1, 0\}$) instead of dense codes (i.e., $\mathbb{M} \in \{1, -1\}$), which are beneficial to efficient training [15].
- using other decoding strategies like loss-based decoding instead of hamming distance to match the codewords of ECOC. Loss-based decoding generally contributes to good performance because of the "confidence" information [15].
- applying low-density parity-check code to design the optimized ECOC.

7 Conclusion

This work investigates class binarization techniques in neuroevolution and proposes the ECOC-NEAT method that applies ECOC to the neuroevolution algorithm of NEAT for multiclass classification. We investigate (1) the performance of NEAT with different class binarization techniques for multiclass classification from multiclass degradation, accuracy, training efficiency, and robustness on three popular datasets, (2) the performance of ECOC-NEAT with different size and quality of ECOC. The results show that ECOC-NEAT offers various benefits compared

to the standard NEAT and NEAT with other class binarization techniques for multiclass classification. Large size ECOCs and optimized ECOCs generally contribute to better performance for multiclass classification. ECOC-NEAT shows significant benefits in a flexible number of binary classifiers and strong robustness. In future, ECOC-NEAT can be extended to other applications such as image classification and computer vision. Moreover, ECOC can be applied to different neuroevolution algorithms for multiclass classification.

Author Contributions G.L.: Conceptualization, Methodology, Validation, Visualization, Investigation, Writing - original draft, Writing - review & editing. Z.G.: Conceptualization, Methodology, Coding and Validation, Visualization, Investigation, Writing - original draft, Writing - review & editing. L.T.: Writing - review & editing. T.L.: Writing - review & editing.

Code and Data Availability The code and data for this work are available at https://github.com/lafengxiaoyu/NEAT-ensembles

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Aly M (2005) Survey on multiclass classification methods. Neural Netw 19:1–9
2. Lorena AC, De Carvalho AC, Gama JM (2008) A review on the combination of binary classifiers in multiclass problems. Artif Intell Rev 30(1):19–37
3. Galar M, Fernández A, Barrenechea E, Bustince H, Herrera F (2011) An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. Pattern Recogn 44(8):1761–1776
4. Zhou J, Peng H, Suen CY (2008) Data-driven decomposition for multi-class classification. Pattern Recogn 41(1):67–76
5. Fürnkranz J (2002) Round robin classification. J Mach Learn Res 2(Mar):721–747
6. McDonnell T, Andoni S, Bonab E, Cheng S, Choi JH, Goode J, Moore K, Sellers G, Schrum J (2018) Divide and conquer: neuroevolution for multiclass classification. In: Proceedings of the genetic and evolutionary computation conference, pp 474–481
7. Floreano D, Dürr P, Mattiussi C (2008) Neuroevolution: from architectures to learning. Evol Intel 1(1):47–62
8. Chen L, Alahakoon D (2006) Neuroevolution of augmenting topologies with learning for data classification. In: 2006 International conference on information and automation. IEEE, pp 367–371
9. Lan G, De Vries L, Wang S (2019) Evolving efficient deep neural networks for real-time object recognition. In: 2019 IEEE symposium series on computational intelligence (SSCI). IEEE, pp 2571–2578
10. Hagg A, Mensing M, Asteroth A (2017) Evolving parsimonious networks by mixing activation functions. In: Proceedings of the genetic and evolutionary computation conference, pp 425–432
11. Lan G, De Carlo M, van Diggelen F, Tomczak JM, Roijers DM, Eiben AE (2021) Learning directed locomotion in modular robots with evolvable morphologies. Appl Soft Comput 111:107688
12. Gao Z, Lan G (2021) A neat-based multiclass classification method with class binarization. In: Proceedings of the genetic and evolutionary computation conference companion, pp 277–278
13. Ng SS, Tse PW, Tsui KL (2014) A one-versus-all class binarization strategy for bearing diagnostics of concurrent defects. Sensors 14(1):1295–1321
14. Adnan MN, Islam MZ (2015) One-vs-all binarization technique in the context of random forest. In: Proceedings of the European symposium on artificial neural networks, computational intelligence and machine learning, pp 385–390
15. Allwein EL, Schapire RE, Singer Y (2000) Reducing multiclass to binary: a unifying approach for margin classifiers. J Mach Learn Res 1(Dec):113–141
16. Liu Y, Yao X (1999) Simultaneous training of negatively correlated neural networks in an ensemble. IEEE Trans Syst Man Cybern Part B (Cybern) 29(6):716–725
17. Abbass HA (2003) Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization. In: The 2003 Congress on Evolutionary Computation, 2003. CEC'03, vol 3. IEEE, pp 2074–2080
18. García-Pedrajas N, Hervás-Martínez C, Ortiz-Boyer D (2005) Cooperative coevolution of artificial neural network ensembles for pattern classification. IEEE Trans Evol Comput 9(3):271–302
19. Stanley KO, Miikkulainen R (2002) Evolving neural networks through augmenting topologies. Evol Comput 10(2):99–127
20. Hsu C-W, Lin C-J (2002) A comparison of methods for multiclass support vector machines. IEEE Trans Neural Networks 13(2):415–425
21. Dietterich TG, Bakiri G (1994) Solving multiclass learning problems via error-correcting output codes. J Artif Intell Res 2:263–286
22. Rokach L (2010) Pattern classification using ensemble methods. World Sci 75:1
23. Bautista MÁ, Escalera S, Baró X, Radeva P, Vitriá J, Pujol O (2012) Minimal design of error-correcting output codes. Pattern Recogn Lett 33(6):693–702
24. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
25. Asuncion A, Newman D (2007) Uci machine learning repository
26. Berger A (1999) Error-correcting output coding for text classification. In: IJCAI-99: Workshop on machine learning for information filtering

27. Verma G, Swami A (2019) Error correcting output codes improve probability estimation and adversarial robustness of deep neural networks. In: Advances in Neural Information Processing Systems, 8646–8656

28. García-Pedrajas N, Ortiz-Boyer D (2011) An empirical study of binary classifier fusion methods for multiclass classification. Inform Fus 12(2):111–130

29. Kong EB, Dietterich TG (1995) Why error-correcting output coding works

30. Grandini M, Bagli E, Visani G (2020) Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.