SODE: Self-Adaptive One-Dependence Estimators for Classification

Jia Wu[a,*], Shirui Pan[a], Xingquan Zhu[b], Peng Zhang[a], Chengqi Zhang[a]

[a] *Quantum Computation & Intelligent Systems (QCIS) Centre, Faculty of Engineering & Information Technology, University of Technology Sydney, NSW 2007, Australia.*
[b] *Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431, USA.*

Abstract

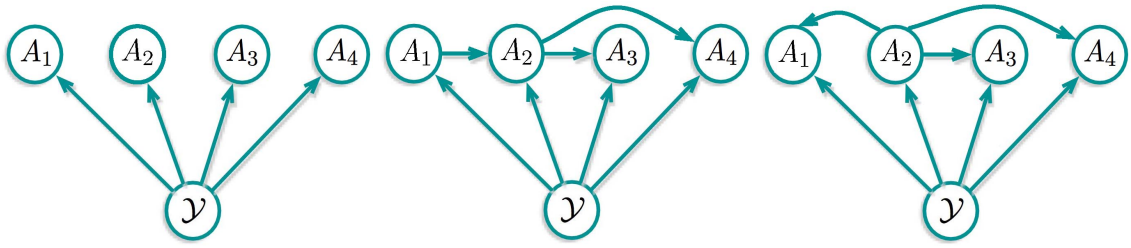
SuperParent-One-Dependence Estimators (SPODEs) represent a family of semi-naive Bayesian classifiers which relax the attribute independence assumption of Naive Bayes (NB) to allow each attribute to depend on a common single attribute (superparent). SPODEs can effectively handle data with attribute dependency but still inherent NB's key advantages such as computational efficiency and robustness for high dimensional data. In reality, determining an optimal superparent for SPODEs is difficult. One common approach is to use weighted combinations of multiple SPODEs, each having a different superparent, by assigning a proper weight value to each superparent (*i.e.,* an attribute). In this paper, we propose a self-adaptive SPODEs, namely SODE, which uses immunity theory in artificial immune systems to automatically and self-adaptively select the weight for each single SPODE. SODE does not need to know the importance of individual SPODE nor the relevance among SPODEs, so it can flexibly and efficiently search optimal weight values for each SPODE during the learning process. Extensive experiments and comparisons on 56 benchmark data sets, and validations on image retrieval and document categorization demonstrate that SODE is suitable for a wide range of tasks and outperforms other state-of-the-art weighted SPODE algorithms. Results also confirm that SODE provides an appropriate balance between runtime efficiency and accuracy effectiveness.

Keywords: Attribute Weighting, Self-Adaptive, Classification, Artificial Immune Systems, Evolutionary Machine Learning

1. Introduction

Naive Bayes (NB) [13] is a simple, efficient, and effective learning algorithm which uses a simplified Bayesian network, as shown in Figure 1(a), with conditional attribute independence assumption for classification [18]. Despite of the strong independence assumption, NB has demonstrated very good classification accuracy, compared to other sophisticated learning methods [51]. Meanwhile, many methods also exist to improve NB by relaxing its attribute interdependence but also retaining its simplicity and efficiency [26, 47, 48, 53, 54]. In this paper, we refer to this type of approaches as semi-naive Bayesian methods.

*Corresponding author. Tel.: +61 416387666, Fax.: +61 2 9514 4535
Email addresses: jia.wu@student.uts.edu.au (Jia Wu), shirui.pan@student.uts.edu.au (Shirui Pan), xzhu3@fau.edu (Xingquan Zhu), peng.zhang@uts.edu.au (Peng Zhang), chengqi.zhang@uts.edu.au (Chengqi Zhang)



(a) NB (b) ODE (c) SPODE

Figure 1: A conceptual view of (a) Naive Bayes (NB), (b) One-Dependence Estimator (ODE), and (c) SuperParent-One-Dependence Estimator (SPODE). Each circle represents an attribute (*e.g.,* class label $\mathcal{Y}$ or attribute $A_i : 1 \leq i \leq 4$). An arrow points from a parent to a child, who only depends on its parents. NB assumes that attributes (A_i) are independent of each other given the class label $\mathcal{Y}$. ODE allows each attribute to depend on at most one other attribute (*i.e.,* parent) in addition to the class. By contrast, SPODE assumes that each attribute depends on a common attribute (*e.g.,* the superparent A_2).

In order to relax the conditional independence assumption in naive Bayes and allow interdependency between attributes, semi-naive Bayesian techniques commonly employ simple wrapper heuristics by minimizing learning error on training data [54]. For example, One-Dependence Estimator (ODE), as an alternative to NB, allows each attribute to depend on at most one other attribute in addition to the class label, as shown in Figure 1(b). Existing analysis and empirical studies [47] have shown that ODE can indeed outperform simple NB when the attribute independence assumption is violated.

SuperParent-One-Dependence Estimator (SPODE), as shown in Figure 1(c), is a subcategory of ODE which allow all attributes to depend on one superparent (*i.e.,* one attribute) in addition to the class label [49]. The employment of a superparent allows SPODE to retain same training efficiency as NB but with a potentially higher classification accuracy. Due to the fact that the superparent plays a major role in SPODE but finding a globally optimal superparent is a challenging task, many existing SPODE methods employ an ensemble based approach by using each single attribute as a superparent to build a SPODE and combining multiple SPODEs for prediction. For example, Averaged One-Dependence Estimators (AODE) [40] combines all SPODEs that satisfy a minimum support constraint and estimate class conditional probabilities by the averaging strategy (*i.e.,* each attribute is treated equally). This approach has demonstrated good classification accuracy with very little extra computational cost. In reality, attributes are playing different roles in learning tasks. A natural way to extend AODE is to assign attributes different weight values, which is the core of weighted SPODEs (WSPODE/WAODE) [22].

In order to discover proper weight values for weighted SPODEs, researchers have proposed many useful methods to evaluate the importance of attributes. Examples include gain ratio [52], correlation-based algorithm [16], mutual information [24], and ReliefF attribute ranking algorithm [34], etc. Although existing attribute weighting SPODEs methods have achieved good performance to solve domain specific problems, all these methods rely on external criteria, such as gain ratio, to determine the weight values of the attributes. In this case, attribute weighting and SPODE learning objective are separated without being considered simultaneously for maximum accuracy gain. To this end, we propose in this paper a new approach to automatically calculate optimal

attribute weight values for SPODEs, by directly targeting SPODEs's objective function. To achieve the goal, we propose to assign proper weight values for weighted SPODEs classification based on immunity theory in artificial immune systems (AIS) [55]. Immune theory has been successfully used to self-adaptively calculate the weight for weighted naive Bayes in previous work [44]. In [45], an immune theory based self-adaptive probability estimation method has also been proposed to select terms and parameters for probability estimation. Therefore, it is appealing to pattern recognition community to have an optimization framework with self-adaptively determined weight values for SPODEs for different learning tasks.

In this paper, we propose to use immune principle to design an automated searching strategy to find optimal attribute weight for each SPODE. The unique immune evolution computation processes, including initialization, clone, mutation, and selection, ensure that our method can adapt to the unique distributions of the underlying data. In contrast to conventional statistical probabilistic evaluation in SPODEs, the proposed immune based SPODEs (SODE) is a self-learning algorithm with immunological properties, such as memory property and clonal selection. To the best of our knowledge, this is the first work to introduce immune principle to the field of SPODE based classification. The niche and advantages of SODE can be understood from the following three aspects:

1) SODE is a data-driven self-adaptive method because it does not requires explicit specification of functional or distributional form for the underlying model or the underlying learning tasks.

2) SODE is a nonlinear model capable of modeling complex real-world relationships.

3) SODE inherits the memory property of human immune systems and can recognize the same or similar antigen quickly at different times.

Our experiments and comparisons on 56 UCI benchmark data sets and validations in image retrieval and document categorization demonstrate that SODE consistently outperforms other state-of-the-art weighted SPODEs algorithms in terms of classification accuracy and variance (*i.e.,* the standard deviation). The runtime comparisons further confirm that SODE provides an appropriate trades-off between learning efficiency and accuracy effectiveness.

The remainder of the paper is structured as follows. Preliminary concepts are addressed in Section 2. Section 3 presents an overview of attribute weighting approaches for SPODE classifiers, followed by a brief review of immune principle in artificial immune systems. Section 4 introduces the proposed algorithm, followed by experiments in Section 5. We conclude the paper in Section 6.

2. Preliminaries

In this section, we introduce important notations and definitions used in the paper.

A training set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1) \cdots, (\boldsymbol{x}_N, y_N)\}$ has N instances, each of which containing n attribute values and a class label. We use $\boldsymbol{x}_i = \{x_{i,1}, \cdots x_{i,j}, \cdots x_{i,n}\}$ to denote the ith instance in the data set $\mathcal{D}$, with $x_{i,j}$ denoting

the jth attribute value and each instance is paired with a class label y_i. The class space $\mathcal{Y} = \{c_1, \cdots, c_k, \cdots, c_L\}$ denotes the set of labels that each instance belongs to and c_k is the kth label of the class space. The attribute space of the data is denoted by $\mathcal{A} = \{A_1, \cdots, A_j, \cdots, A_n\}$, where A_j denotes the jth attribute. Each attribute can be a discrete random variable (with a number of discrete values) or a continuous random variable. In this paper, we only focus on categorical (or nominal) attributes, and for any attribute A_j, we use $a_j^\tau, \tau = 1, \cdots, |A_j|$ to denote the τth attribute value of A_j and $|A_j|$ denotes the total number of distinct values of A_j. For each instance $\boldsymbol{x}_i$, its value satisfies $x_{i,j} \in A_j$.

SPODE-based Classifiers: For an instance $(\boldsymbol{x}_i, y_i)$ in the training set D, its class label satisfies $y_i \in \mathcal{Y}$, whereas a test instance $\boldsymbol{x}_t$ only contains attribute values and its class label y_t needs to be predicted by the classification model.

A Maxim A Posteriori (MAP) classifier aims to determine the class label of a test instance $\boldsymbol{x}_t$ by maximizing the posteriori probability as follows:

$$c(\boldsymbol{x}_t) = \arg\max_{c_k \in \mathcal{Y}} P(c_k|\boldsymbol{x}_t) = \arg\max_{c_k \in \mathcal{Y}} P(\boldsymbol{x}_t, c_k) \tag{1}$$

Since $P(c_k|\boldsymbol{x}_t)$=$P(c_k, \boldsymbol{x}_t)/P(\boldsymbol{x}_t)$ and $P(\boldsymbol{x}_t)$ is invariant across different class labels, one only needs to calculate $P(c_k, \boldsymbol{x}_t)$ to determine the final class label. In reality, when the number of training samples is limited, the estimation of joint distribution $P(c_k, \boldsymbol{x}_t)$ is usually unreliable. Therefore, approximating $P(c_k, \boldsymbol{x}_t)$ becomes the key challenge of deriving Bayesian learning models [23].

2.1. Naive Bayes

In reality, because joint probability $P(c_k, \boldsymbol{x}_t) = P(\boldsymbol{x}_t|c_k) \times P(c_k)$, a straightforward approach to simplify the joint probability estimation is to simply ignore the dependency relationships between attributes and assume all attributes are conditionally independent, given the class label c_k. By doing so, the probability of observing the conjunction of all attributes is simplified as the product of the probabilities of each individual attributes. This assume results in the core concept of naive Bayes (NB) as follows,

$$c(\boldsymbol{x}_t) = \arg\max_{c_k \in \mathcal{Y}} \prod_{j=1}^{n} P(x_{t,j}\,|c_k) P(c_k) \tag{2}$$

In naive Bayes, each attribute node only has a class node as its parent, which makes the learning highly efficient. However, such settings may also reduce the classification performance due to the ignorance of attribute interdependency.

2.2. SuperParent-One-Dependance Estimators

To improve NB classification, semi-naive Bayesian approaches are proposed to exploit attribute dependencies at a moderate degree. For example, k-dependence estimator (k-DE) [37] allows each attribute to have the class $\mathcal{Y}$ and a maximum of k other attributes as parents (*i.e.* a naive Bayes (NB) classier is a 0-dependence estimator).

Among all existing semi-naive Bayesian approaches, One-Dependence Estimator (ODE, *e.g.,* Tree Augmented Naive Bayes (TAN) [15]) based methods have achieved good trade-off between classification efficiency and effectiveness [47]. For ODEs, each attribute is allowed to depend on at most one other attribute in addition to the class label.

SuperParent-One-Dependence Estimators (SPODEs) are a special type of ODEs which require all attributes to depend on the same attribute, *i.e.,* the superparent [49], as shown in Figure 1(c). Indeed, SPODEs offer a good combination of training efficiency, classification efficiency, and accuracy [40, 22, 48, 49]. A SPODE with superparent A_q will estimate the probability of each class label c_k given an instance $\boldsymbol{x}_t$ as follows (where $x_{t,q}$ denotes the value of attribute A_q in instance $\boldsymbol{x}_t$):

$$\begin{aligned} c(\boldsymbol{x}_t) &= \arg\max_{c_k \in \mathcal{Y}} P(c_k, \boldsymbol{x}_t) \\ &= \arg\max_{c_k \in \mathcal{Y}} P(c_k, x_{t,q}) P(x_t | c_k, x_{t,q}) \\ &= \arg\max_{c_k \in \mathcal{Y}} P(c_k, x_{t,q}) \prod\nolimits_{j=1}^{n} P(x_{t,j} | c_k, x_{t,q}) \end{aligned} \tag{3}$$

2.3. Averaged One-Dependence Estimators

The first approach to use SPODEs for learning is Averaged One-Dependence Estimators (AODE) [40], which uses an average ensemble of all SPODEs. In [40], this simple selection criterion has shown good performance in prediction accuracy and runtime efficiency, compared to TAN [15], as

$$c(\boldsymbol{x}_t) = \arg\max_{c_k \in \mathcal{Y}} \left(\sum_{1 \le q \le n} P(c_k, x_{t,q}) \prod\nolimits_{j=1}^{n} P(x_{t,j} | c_k, x_{t,q}) \right) \tag{4}$$

On the other hand, there are also some subsequent SPODE ensemble strategies, such as MAPLMG [8] scheme (maximum a posteriori linear mixture of generative distributions) and BMA (Bayesian model averaging) [19]. Moreover, Yang *et al.* [47] proposed a forward sequential addition strategy to iteratively add the SPODEs based on the hill-climbing search method.

2.4. Attribute Weighted SPODE

In AODE, each single SPODE is treated equally, which essentially means that attributes are treated equally. In real-world applications, attributes play different roles in classification. A natural way to extend AODE is to assign different weight values to attributes, which is the design of the WSPODE [22]:

$$c(\boldsymbol{x}_t) = \arg\max_{c_k \in \mathcal{Y}} \left(\sum_{q=1}^{n} w_q P(c_k, x_{t,q}) \prod_{j=1}^{n} P(x_{t,j} | c_k, x_{t,q}) \right) \tag{5}$$

where w_q is the weight of the SPODE for attribute A_q.

By proposing to use immune principle to search optimal weight values for attribute weighted SPODE, our method is related to attribute weighting in machine learning and immune evolutionary computation.

3. Related Work

3.1. Attribute Weighted Methods

In real-world applications, attributes often play different roles. Assigning different weight values to attributes is potentially helpful in improving the classification performance. In this subsection, we review existing work on attribute weighting by separating them into two main categories: methods considering each single attribute's correlation to the class, and methods considering multiple attributes' joint correlations to the class.

3.1.1. Single Attribute Correlation Weighting

Mutual Information (MI) provides a quantitative measure to evaluate the mutual dependence of two variables. A high MI value indicates a large reduction of uncertainty of one random variable, after observing another random variable, and therefore suggests a strong correlations between two random variables. A zero MI value between two random variables means the variables are independent. Mutual information has a long history of being used for measuring correlations between attributes and the class variable in classification. For instance, Jiang & Zhang [24] applied this method to improve the accuracy for AODE.

Gain Ratio (GR) is used to solve the drawback by dividing each attribute's IG (Information Gain) score by the information encoded in each attribute itself. It has been commonly used to evaluate the correlation of attributes to the class for decision tree learning. A notable drawback of IG is that the resulting score is biased to attributes with a large number of distinct values, and common solutions are to divide IG scores by the entropy of each attribute, resulting in Information Gain Ratio measure. In [52], Zhang & Sheng proposed to assign a higher weigh value to attributes with a larger gain ratio value in weighted naive Bayes (WNB).

3.1.2. Multiple Attribute Correlation Weighting

Correlation-based Feature Selection (CFS) for attribute weighing uses a correlation-based heuristic evaluation function as an attribute quality measure to calculate the weight value of each attribute. It uses a best-first search to traverse the feature space. CFS starts with an empty set and generates all possible single feature expansions. The subset with the highest evaluation is selected and expanded in the same manner by adding new features. If expanding a subset results in no improvement, the search drops back to the next best unexpanded subset and continues from there. The best subset found is returned after the search terminates. The core of CFS is the heuristic process that evaluates the worth or "merit" of a feature subset. Hall [16] employed this method to evaluate the importance of attributes according to the heuristic "merit" value.

Relief-F is a feature selection method based on attribute estimation [34]. Relief-F assigns a grade of relevance to each feature by examining the change of the feature values with respect to instances within the same class (*i.e.,* the nearest hit) and instances between classes (*i.e.,* the nearest miss). If a feature's values remain relatively stable

for instances within the same class, the feature will receive a higher weight value. Wu *et.al.* [44] applied Relief-F attribute weighted approach to calculate the attribute for WNB.

Attribute Correlation-based Weighting explicitly considers the correlation of each attribute to all other attributes to calculate the attribute's weight value [16]. A large weight value will be assigned to the attributes with strong dependencies on other attributes. In order to estimate each attribute's dependence, an unpruned decision tree is constructed from the training instances with a minimum *depth*, which indicates the depth for testing the tree. The weight assigned to each attribute is inversely proportional to the minimum depth at which they were first tested in an unpruned decision tree. Attributes that do not appear in the tree receive a zero weight value. In [43], this type of approach has been proposed as a state-of-art weighting to enhance the performance of AODE.

Maximum a Posteriori Linear Mixture of Discriminative Distributions (MAPLMD) is proposed to improve the performance of the Bayesian Model Averaging (BMA) [19, 12, 30], a well used coherent framework for the purpose of integrating learning models to solve the uncertainty problem when using a single model. BMA has been successfully applied to weighted SPODEs in [8]. However, it has been proved that BMA cannot provide a better approximation than AODE to their probability distributions most of the times [48, 49]. In order to carry out the exact BMA prediction, a straightforward solution is to use MarKov Chain Monte Carlo (MCMC) for the approximation. However, such designs are subject to expensive computational costs. Accordingly, MAPLMD first constructs a linear mixture of discriminative distribution model, and then determines the weight by using Expectation-Maximization (EM) method for Maximum A Posteriori (MAP) estimation.

Maximum a Posteriori Linear Mixture of Generative Distributions (MAPLMG) finds the best weight for an ensemble of generative distribution model by maximizing the supervised posterior probability. The maximization problem in a linear mixture of generative distribution model is a constrained nonlinear optimization issue, which can be solved by adopting the augmented (or penalized) lagrangian approach [1]. By adjusting the penalization provided by not fulfilling the constraints, the constrained nonlinear optimization issue can be transformed into a sequence of unconstrained optimization problems (*i.e.,* means of a sequence of unconstrained maximizations) [8], each of which is solvable by the well known Newton-like procedure, Broyden-Fletcher-Goldfarb-Shanno (BFGS) [29].

3.2. Immune Principle

The immune principle in Artificial Immune Systems (AIS) consists of three major components, including representation, recognition, and clone selection, as shown in Figure 2. The representation, known as shape-shape problem, focuses on the modeling of antibodies and antigens. When the immune system is attacked by antigen (*i.e.* foreign substances), antibodies try to neutralize the infection by binding to the antigen through a recognition process. Binding strength, also regarded as affinity, is used as a threshold for the immune system to respond to the antigen. The clone selection is corresponding to an affinity maturation process, which means that immune individuals with low affinity will gradually increase during clone and mutation process. Meanwhile,

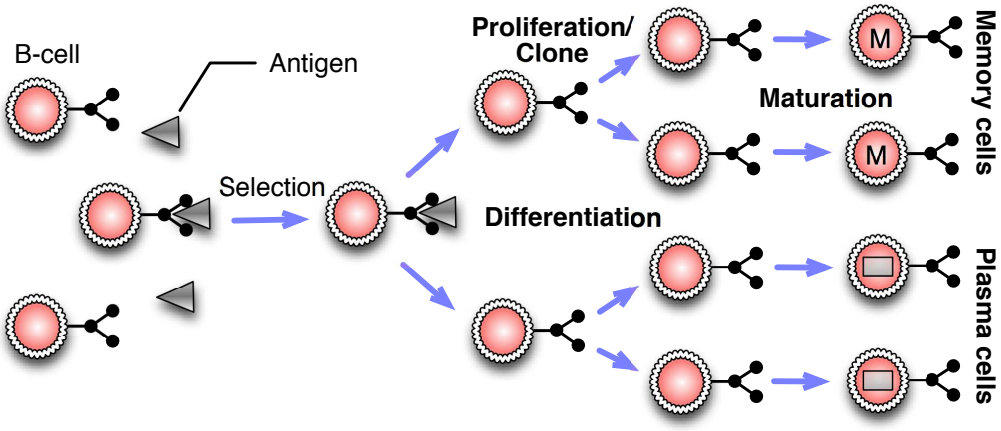


Figure 2: A concept view of immune principle: A B-cell contains the antibody (the middle rings on the left) that allows it to recognize the antigen (triangle), which denotes pathogenic materials invading to the system. The binding between B-cell and antigen can be evaluated by using certain affinity (*i.e.,* degree of binding). In a learning system, this resembles to the assessment of how good a solution (*i.e.,* antibody) recognize/resolve the training data (*i.e.,* antigen). After recognition, the system will respond and result in proliferation, differentiation and maturation process of the B-cell as secondary antibodies. The secondary antibodies with high affinity becomes a memory cell, and others become plasma cells. The memory cells are retrained in the system to allow faster response to the same (or similar) attacks in the future (if the body is re-infected by the same pathogenic materials).

some immune individuals will polarize into memory individuals, which will be propagated to the future iterations.

Similar to the AIS, evolutionary algorithms (EAs) [31], such as Genetic Algorithms (GA) [36], Evolution Strategies (ES) [3] and Differential evolution (DE) [38] are all designed based on biological evolution to control and optimize artificial systems. An immune system has a mechanism of memorizing past events to continually improve the learning for any new encounters. Moreover, AIS is highly distributed, highly adaptive, and self-organising. In addition, AIS is also a general framework for distributed systems, which can be easily applied to many domains. Because of its self-organizing nature, AIS typically requires very few learning parameters. Evolutionary computation and AIS share striking similarities in many key concepts, such as populations and proliferation of individuals mostly fit to the environment. Some previous works have pointed out the similarities and the differences between immune principle in AIS and other heuristics [55, 32, 7]. From the intuition perspective, EAs are inspired by natural evolution, whereas AIS is inspired by the natural immune system, with the clonal principle as a basic and important mechanism. The mutation in evolution is random, whereas the hypermutation process of clonal selection in AIS is controlled and directly proportional to the receptors affinity with the triggering antigen. In [7], the authors suggested that works on EAs can be leveraged by AIS, which indicates that research on selection operations (*e.g.,* tournament, roulette, wheel, etc.) may be exploited. AIS can reach a diverse set of local optima solutions, while the EAs tend to bias the whole population of individuals toward the best candidate solution. Essentially, their encoding schemes and evaluation functions are similar, but their evolutionary search processes differ from many key aspects, such as inspiration, vocabulary, and sequence of steps. In summary, evolutionary algorithms utilize a vocabulary borrowed from natural genetics and are inspired in the Darwinian evolution, by the contrast, AIS algorithms adopt the shape-space formalism, along with

immunological terminology to describe antibody and antigen interactions and cellular evolution [6].

Immune mechanism has been used in many applications [50, 28, 42, 56, 35, 33], including pattern recognition, clustering, and optimization, etc. Furthermore, the immune theory has been successfully employed to calculate weight for weighted NB [44] and self-adaptive probability estimation for Bayesian learning [45]. In this paper, we propose an immune strategy based adaptive weighting method to improve weighted SPODEs. It is worth noting that some works exist to improve AIS to solve domain specific problems, such as an improved artificial immune system for seeking the Pareto front of land-use allocation problem in large areas [21]. However, in this paper the improved AIS for weighted SPODEs is not included, mainly because that we aim at proposing a general self-adaptive weighting framework for weighted SPODEs which can be also generalized to other improved AIS algorithms.

4. SODE: Self-adaptive SPODE

4.1. Problem Definition

In this paper, we aim to search optimal attribute weight values for each SPODE, so all SPODEs can be combined to form an accurate classifier. Notice that although many approaches exist to determine attribute weight values for classification, our problem differs from them by using a combined objective function which unifies the search of the optimal weight values and the SPODE learning into a single learning process. Assume that each SPODE has an optimal w_q value, and n SPODEs are combined to form a combined classifiers, there are n weight values w_q values needed to be found during the classification process. Therefore, the weighted SPODE classification can be translated to an optimization problem as follows.

$$\mathbf{w}^* = \arg\max_{w_q \in \mathbf{w}} f(\boldsymbol{x}_t, \mathbf{w}) \quad s.t. \;\; 0 \leq w_q \leq 1 \tag{6}$$

where $\mathbf{w} = \{w_1, \cdots, w_q, \cdots, w_n\}$ denotes the attribute weight vector for all SPODEs. $f(\boldsymbol{x}_t, \mathbf{w})$ is calculated by Eq. (5).

4.2. Weight Optimization for SODE

By proposing to use the immune theory in artificial immune systems to search optimal weight values for weighted SPODE classification, our method is related to attribute weighting in machine learning and immune evolutionary computation. In our solution, antigens in SODE are simulated as samples or training data which are presented to the system during the training and the testing process. The antibody as candidate, presented by attribute weight vector $\mathbf{w}$ which has good affinity, will experience a form of clonal expansion after being presented with input data sets (analogous to antigens). When antibodies are cloned they will undergo a mutation process, in which specific mutation function will be designed. The evolving optimization process of the immune system will help discover optimal $\mathbf{w}$ vector with the best classification accuracy. Before introducing algorithm details, we briefly define following key notations, which will help understand the learning of the weight values

Table 1: Symbol Mapping between Immune System and SODE.

Immune system	SODE
Antibody	Attribute weight vector $\mathbf{w}$.
Antigens	A set of samples provided for learning $\mathcal{D}^a$, (80% of training set $\mathcal{D}$ in our experiments).
Shape-space	Possible values of the data vectors.
Affinity	The accuracy of the classifier built from $\mathcal{D}^a$ by using the weight vector $\mathbf{w}$, and validated on $\mathcal{D}^b$ (20% of training set $\mathcal{D}$ in our experiments).
Clonal Expansion	Reproduction of weight vectors $\mathbf{w}$ that are well matched with antigens.
Affinity Maturation	Specific mutation of $\mathbf{w}$ vector, including the removal of lowest stimulated weight vectors.
Immune Memory	Memory set of mutated weight vectors.

using immune principle. In Table 1, we summarize the mapping of the symbols between immune system and SODE.

- *Antibodies:* $\mathcal{W}$ represents the set of antibodies, $\mathcal{W} = \{\mathbf{w}_1, \cdots, \mathbf{w}_L\}$, where L represents the size of antibodies. $\mathbf{w}_i = \{w_{i,1}, \cdots w_{i,j}, \cdots w_{i,n}\}$ represents a single antibody (*i.e.,* attribute weight vector). So $w_{i,j}$ will represent the jth value of the ith antibody $\mathbf{w}_i$.

- *Antigens:* $\mathcal{D}^a$ represents the set of antigens, $\mathcal{D}^a = \{\boldsymbol{x}^a_1, \cdots, \boldsymbol{x}^a_{N_a}\}$, where N_a represents the size of antigens. $\boldsymbol{x}^a_i$ represents a single antigen. In SODE, Antigens resemble to the samples which are provided to help build the leaning models. So $\boldsymbol{x}^a_i$ denotes an instance in the data set $\mathcal{D}^a$.

- *Affinity:* A measure of fitness/closeness between antibodies and antigens. In the current implementation, this value is calculated as accuracy on given data set $\mathcal{D}^b = \{\boldsymbol{x}^b_1, \cdots, \boldsymbol{x}^b_{N_b}\}$ with N_b instances.

- *Memory Cell:* $\mathbf{w}_c$ represents the memory cell for the antibody which has the best affinity (*i.e.,* best accuracy on $\mathcal{D}^b$).

- *Clone rate:* An integer value used to determine the number of mutated clones for a given antibody (*i.e.,* attribute weight vector).

- *Mutation rate:* A parameter between 0 and 1 that indicates the probability that an antibody is mutated. For a given antibody, 1 minus its affinity will be considered as the resulting mutation rate. So, the antibody with high affinity will receive low mutation probability.

The overall framework of the proposed SODE includes the following two major steps: (1) Using immune strategy to determine the optimal weight values for each single SPODE classifier (*i.e.,* weight optimization as shown in Figure 3); and (2) Classifying each test instance using SODE with the optimal weight. The detailed process is described as follows:

Initialization For individuals in $\mathcal{W} = \{\mathbf{w}_1, \cdots, \mathbf{w}_L\}$ with its population size L, we ensure that every individual $\mathbf{w}_i = \{w_{i,1}, \cdots w_{i,j}, \cdots w_{i,n}\}$ in antibody population is generated through a random mechanism, by setting $w_{i,j}$ of $\mathbf{w}_i$

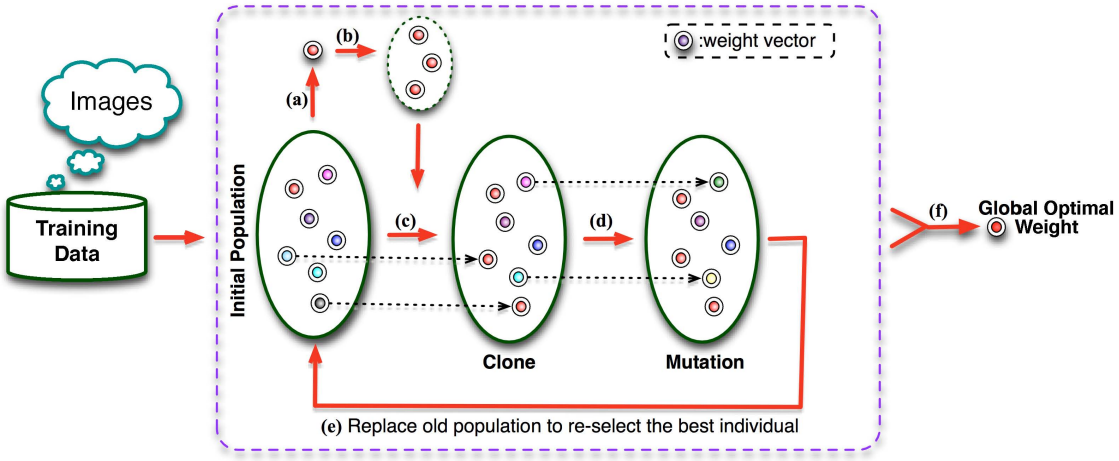


Figure 3: A conceptual view of self-adaptive weighting strategy for SODE: An initial population contains many antibodies (*i.e.,* weight vectors $\mathbf{w}$) that allow themselves to recognize antigens (*i.e.,* $\mathcal{D}^a$) with certain affinity (*i.e.,* accuracy on $\mathcal{D}^b$ via the classier build on $\mathcal{D}^a$ with weight vectors $\mathbf{w}$; $\mathcal{D}^a \cup \mathcal{D}^b = \mathcal{D}$ (training set)). After recognition, the system will respond and select the weight vector $\mathbf{w}_c^t$ with the best affinity (a), and then clone it (b) to replace the weight vectors with low affinity (c). After that, mutation strategy is adopted to maintain the diversity of the weight vectors (d). The mutation population will further replace the old population (e) to reselect the best weight vector as the memory antibody (a). Through the evolutionary process, the search will aim to find global optimal weigh vector $\mathbf{w}^*$ (f) to build weighted SPODE classifier-SODE.

a uniformly distributed random number within range $[0, 1]$. In order to evaluate the fitness of a model, we use a set of training instances $\mathcal{D}$ as antigens $\mathcal{D}^a$, and the remaining instances in $\mathcal{D}$ are used as $\mathcal{D}^b$ to assess the $\mathbf{w}_c$ (*i.e.,* memory antibody). In our experiments, we set $\mathcal{D}^a$ as 80% of training data set $\mathcal{D}$, so $\mathcal{D}^b$ is 20% of the training set $\mathcal{D}$. L is set to 50, which is the same used in [44].

Weight Evaluation

- **Calculation of affinity function:** The affinity of the ith individual of the tth generation $\mathbf{w}_i^t$ is the classification accuracy that is obtained by SODE trained from $\mathcal{D}^a$ with $\mathbf{w}_i^t$ to carry out the probability estimation. Calculation of affinity function can be described as

$$f[\mathbf{w}_i^t] = \frac{1}{N^b} \sum_{i=1}^{N^b} \delta[\mathrm{c}(\boldsymbol{x}_i^b), y_i^b] \tag{7}$$

  where, $c(\boldsymbol{x}_i^b)$ is the classification result of the ith instance in $\mathcal{D}^b$ with N_b instances, using the SODE trained on $\mathcal{D}^a$ with individual $\mathbf{w}_i^t$. y_i^b is the true class value of $\boldsymbol{x}_i^b$. $\delta[\mathrm{c}(\boldsymbol{x}_i^b), y_i^b] = 1$ if $c(\boldsymbol{x}_i^b) = y_i^b$ and zero otherwise.

- **Antibody Clone:** The individual $\mathbf{w}_c^t$ with the best affinity will be selected as the memory antibody to be further cloned. To ensure the population size of every generation is fixed, $\mathbf{w}_c^t$ will be cloned under the clone factor c to replace the individuals in $\mathcal{W}$ with low affinity under the same rate c.

- **Antibody Mutation:** Applying mutation to the individuals in the tth generation $\mathcal{W}^t$, to ensure the diversity of the antibodies. It means that we obtain the generation composed with the new variation individuals from the parent generation. For any individual $\mathbf{w}_i^t$ from the tth generation, the new variation individual $\mathbf{v}_i^{t+1}$ can

be generated as follows:

$$\mathbf{v}_i^{t+1} = \mathbf{w}_i^t + F * N(0,1) * (\mathbf{w}_c^t - \mathbf{w}_i^t) \tag{8}$$

Among them, N(0,1) is a normally distributed random variable within the range [0,1]. $F = 1 - f[\mathbf{w}_i^t]$, as the variation factor during the process of evolution, can be adaptively obtained according to the different clones [44]. $f[\mathbf{w}_i^t]$ denotes the affinity of the ith individual of the tth generation. In this case, the antibody with high affinity will have a low probability being mutated. As a result, it will accelerate the affinity maturation.

- **Antibody Crossover:** After obtaining mutation antibodies, the crossover operation will be used between the individual $\mathbf{w}_i^t$ and its corresponding variation individual $\mathbf{v}_i^{t+1}$ to generate crossover individuals $\mathbf{c}_i^{t+1}$. By doing so, some new individuals with high affinity may be generated to approach to the optimal solution. The antibody crossover can be formulated as:

$$c_{i,j}^{t+1} = \begin{cases} v_{i,j}^{t+1}, rand(j) \leq F \text{ or } j = randn(i) \\ w_{i,j}^{t}, rand(j) > F \text{ and } j \neq randn(i) \end{cases} \tag{9}$$

where $rand(j) \in [0,1]$ is a uniformly distributed random number, with j denoting the index of the individual. The parameter F, which is the same used in Eq. (8), is used to determine which dimension of the individual $\mathbf{w}_i^t$ will be replaced by the variation individual $\mathbf{v}_i^{t+1}$. In this case, the individual with high affinity will receive a small crossover probability to maintain its good performance. In addition, $randn(i)$ is denoted as a random integer between $[1, n]$ to ensure that at least one dimension variable of the individual $\mathbf{w}_i^t$ is contributed by the variation vector $\mathbf{v}_i^{t+1}$. Otherwise, individual $\mathbf{w}_i^t$ and the crossover individual $\mathbf{c}_i^{t+1}$

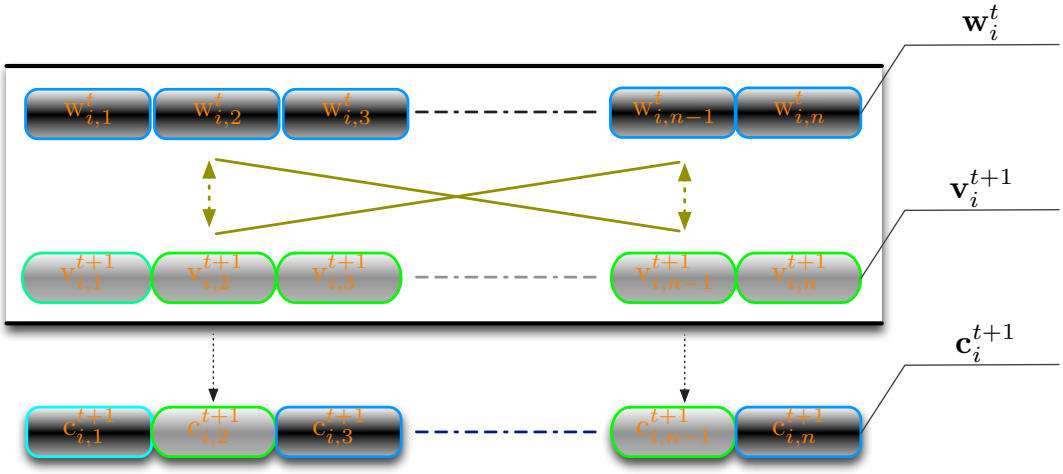


Figure 4: An example of the crossover procedure used in SODE. For a n-dimensional vector (*i.e.,* attribute weight individual) $\mathbf{w}_i^t$ in the ith generation, its crossover vector $\mathbf{c}_i^{t+1}$ is obtained by reorganizing some values using the variation individual $\mathbf{v}_i^{t+1}$ under the cross mechanism in Eq. (9).

Algorithm 1 SODE (Self-Adaptive SPODEs)

Input:

Clone Factor c, Threshold T, Training Set $\mathcal{D}$;
Maximum Iterations $MaxGen$,Antibody Population $\mathcal{W}$;

Output:

The target class label $c(\boldsymbol{x}_t)$ of test instance $\boldsymbol{x}_t$;

$\mathcal{W}, \mathcal{D}^a, \mathcal{D}^b \leftarrow$ Initialize $\mathcal{W}, \mathcal{D}^a$, and $\mathcal{D}^b$ via **Initialization**
while $t \leq MaxGen$ *and* $f[\mathbf{w}_c^{t+1}] - f[\mathbf{w}_c^t] \geq T$ **do**
  $f[\mathbf{w}_i^t] \leftarrow$ Apply $\mathcal{D}^a$ and $\mathcal{D}^b$ to antibody $\mathbf{w}_i^t$, and calculate the affinity of $\mathbf{w}_i^t$.
  $\mathbf{w}_c^t \leftarrow$ Apply the sequence of each $f[\mathbf{w}_i^t]$ and find the $\mathbf{w}_c^t$ with the best affinity.
  $(\mathcal{W}^r)^t \leftarrow$ Select antibody set with the lowest affinity with clone factor c.
  $(\mathcal{W}^c)^t \leftarrow$ Clone $\mathbf{w}_c^t$ with clone factor c and obtain clone antibody set.
  $\mathcal{W}^t \leftarrow [\mathcal{W}^t - (\mathcal{W}^r)^t] \cup (\mathcal{W}^c)^t$;
  for all each $\mathbf{w}_i^t$ in $\mathcal{W}^t$ **do**
    $\mathbf{v}_i^{t+1} \leftarrow$ Apply $\mathbf{w}_c^t$ to $\mathbf{w}_i^t$ and obtain the mutation individual by using Eq. (8).
    $\mathbf{c}_i^{t+1} \leftarrow$ Apply $\mathbf{v}_c^t$ to $\mathbf{w}_i^t$ and obtain the crossover individual by using Eq. (9).
    $\mathbf{w}_i^{t+1} \leftarrow$ Apply $\mathbf{c}_i^{t+1}$ to $\mathbf{w}_i^t$ and obtain the new individual in the $t+1$th generation.
  end for
end while
$\mathbf{w}^* \leftarrow \mathbf{w}_c$; *// Global optimal weight*
$c(\boldsymbol{x}_t) \leftarrow$ Apply $\mathbf{w}^*$ to instance $\boldsymbol{x}_t$ to predict its class label.

will be the same, which brings no help to the evolution process. The process of the crossover operation for an n-dimensional variable is described in Figure 4.

Update of Weight In order to determine whether the crossover individual $\mathbf{c}_i^{t+1}$ can replace the target individual vector $\mathbf{w}_i^t$ to be the new individual $\mathbf{w}_i^{t+1}$ in the $t+1$th generation, a greedy search strategy is employed. If the affinity of $\mathbf{c}_i^{t+1}$ is better than that of the target individual $\mathbf{w}_i^t$, it will be chosen as the offspring. The system chooses the individual $\mathbf{w}_c^{t+1}$ with the best affinity performance in the $t+1$th generation as the new memory antibody.

Classification An unabridged evolutionary process for the population includes **Evaluation** and **Update**, which continuously repeats until (1) the algorithm surpasses the pre-set maximum number $MaxGen$; or (2) the result gap obtained from two consecutive iterations is less than the threshold (*i.e.,* T). The resulting SPODEs classier with optimal weight vector $\mathbf{w}^*$ will further used to predict the test sample $\boldsymbol{x}_t$. The detailed SODE is summarized in Algorithm 1.

5. Experiments

5.1. Experimental Setting

We implement the proposed method using WEKA [41] data mining tool. Because SPODE based classifiers are designed for categorical attributes, in our experiments, we first replace all missing attribute values using unsupervised attribute filter *ReplaceMissingValues* in WEKA. Then, we apply unsupervised filter *Discretize* in WEKA to discretize numeric attributes into nominal attributes. Similar data preprocessing approaches can also be found in the previous works [44, 45]. In our experiments, the algorithm performances are evaluated

in terms of classification accuracy, standard deviation, time complexity, and CPU runtime. Besides, the three parameters maximum iteration *MaxGen*, threshold T, and the clone factor c in Algorithm 1 are set to 50, 0.001 and 0.1 respectively. All reported results are based on 10 runs of 10-fold cross validation, and all experiments are conducted on a Linux cluster node with an Interl(R) Xeon(R) @3.33GHZ CPU and 3GB fixed memory size.

5.2. Baseline Methods

We continue by introducing baselines and their abbreviations. Because the proposed SODE is a weighted SPODE approach, so we use the existing weighted SPODEs as baselines. For other semi-naive Bayesian classifiers (*e.g.,* NB, ODE, *et. al.*), existing research [40] has systematically validated and demonstrated that they are inferior to AODE. Therefore, we only compare the proposed algorithm with AODE, but not include the comparisons with NB or semi-naive NB, such as ODE *et. al*.

1. *AODE* : The AODE classifier by using average of multiple SPODEs [40].
2. *CODE* : Weighted SPODE based on correlation-based feature selection [17].
3. *GODE* : Weighted SPODE based on gain ratio [52] for feature weighting.
4. *MODE* : Weighted SPODE using mutual information based feature weighing method [22].
5. *RODE* : Weighted SPODE using a Relief-F attribute ranking based feature estimation [34].
6. *TODE* : Weighted SPODE with the weighting method according to the degree to which they depend on the values of other attributes [43].
7. *DODE* : Weighted SPODE with the weighting method based on maximizing a posteriori linear mixture of discriminative distributions (MAPLMD) [49].
8. *PODE* : Weighted SPODE with the weighting method based on maximizing a posteriori linear mixture of generative distributions (MAPLMG) [48].
9. **SODE** : The proposed self-adaptive weighting SPODE to dynamically calculate the weight value.

For all the above methods, the probability values $P(c_k, x_{t,q})$ and $P(x_{t,j}|c_k, x_{t,q})$ for SPODE in Eq. (3) are estimated by using the Laplace estimate as

$$p(c_k, x_{t,q}) = \frac{F(c_k, x_{t,q}) + 1.0}{\mathrm{N} + \mathrm{L}} \tag{10}$$

$$P(x_{t,j}|c_k, x_{t,q}) = \frac{F(c_k, x_{t,q}, x_{t,j}) + 1.0}{F(c_k, x_{t,q}) + |A_j|} \tag{11}$$

where, $|A_j|$ is the number of distinct values (*e.g.,* $x_{t,j}$) of attribute A_j and L is the number of classes in the training data. $F(\cdot)$ is the frequency with which a combination of terms with N denoting the number of training samples.

Table 2: Data characteristics of the benchmark data

Data Set	Instances	Attributes	Classes	Missing	Numeric
anneal	898	39	6	Y	Y
anneal.ORIG	898	39	6	Y	Y
artificial-characters	10218	8	10	N	Y
audiology	226	70	24	Y	N
autos	205	26	7	Y	Y
balance-scale	625	5	3	N	Y
breast-cancer	286	10	2	Y	N
breast-w	699	10	2	Y	N
car	1728	7	4	N	N
climate	540	21	2	N	Y
colic	368	23	2	Y	Y
colic.ORIG	368	28	2	Y	Y
credit-a	690	16	2	Y	Y
credit-g	1000	21	2	N	Y
cylinder-bands	540	41	2	Y	Y
diabetes	768	9	2	N	Y
ecoli	336	8	8	N	Y
energy-y1	768	9	37	N	Y
energy-y2	768	9	38	N	Y
Glass	214	10	7	N	Y
hayes-roth	160	5	3	N	Y
heart-c	303	14	5	Y	Y
heart-h	294	14	5	Y	Y
heart-statlog	270	14	2	N	Y
hepatitis	155	20	2	Y	Y
hypothyroid	3772	30	4	Y	Y
ionosphere	351	35	2	N	Y
iris	150	5	3	N	Y
kr-vs-kp	3196	37	2	N	N
labor	57	17	2	Y	Y
letter	20000	17	26	N	Y
lymph	148	19	4	N	Y
mfeat-f	2000	77	10	N	Y
monks	556	7	2	N	Y
movement-libras	360	91	15	N	Y
mushroom	8124	23	2	Y	N
newthyroid	215	6	3	N	Y
optdigits	5620	63	10	N	Y
page-blocks	5473	11	5	N	Y
pendigits	10992	17	10	N	Y
primary-tumor	339	18	21	Y	N
qar-biodegradation	1055	42	2	N	Y
robot-24	5456	25	4	N	Y
segment	2310	20	7	N	Y
sick	3772	30	2	Y	Y
sonar	208	61	2	N	Y
soybean	683	36	19	Y	N
spectrometer	531	102	48	N	Y
splice	3190	62	3	N	N
steel-plates-faults	1941	34	2	N	Y
texture	5500	41	11	N	Y
vehicle	846	19	4	N	Y
vote	435	17	2	Y	N
vowel	990	14	11	N	Y
waveform	1000	41	3	N	Y
zoo	101	18	7	N	Y

Table 3: Classification accuracy comparisons on UCI data sets (%).

Data Set	**SODE**	AODE [40]	PODE [48]	DODE [49]	CODE [17]	GODE [52]	MODE [24]	RODE [34]	TODE [43]
anneal	98.41	97.15 •	97.05 •	96.97 •	96.92 •	96.90 •	97.38 •	97.54	96.83 •
anneal.ORIG	90.02	89.01 •	89.52	89.37 •	89.40	89.59	89.70	89.86	89.59
artificial-characters	58.78	56.62 •	57.87 •	58.09	56.31 •	56.79 •	56.83 •	57.55 •	56.86 •
audiology	75.96	75.96	71.66 •	71.61 •	71.70 •	71.66 •	71.61 •	71.57 •	71.66 •
autos	79.81	74.60 •	75.08 •	76.86 •	75.18 •	75.08 •	75.43 •	75.32 •	74.60 •
balance-scale	89.45	89.78	89.25	89.49	88.32	89.65	89.65	89.15	89.71
breast-cancer	72.18	72.73	72.18	72.56	72.77	71.80	72.25	72.53	72.67
breast-w	96.71	96.85	96.95	96.88	96.68	96.82	96.82	96.81	96.67
car	94.11	91.41 •	92.29 •	92.30 •	91.30 •	90.39 •	90.75 •	91.04 •	91.19 •
climate	90.93	88.43 •	88.94 •	88.93 •	87.96 •	87.48 •	87.46 •	87.54 •	88.28 •
colic	81.26	80.93	81.01	81.45	81.01	81.36	81.50	81.72	81.23
colic.ORIG	77.71	75.38	75.87	75.06	75.60	75.71	76.26	75.82	75.85
credit-a	84.93	85.86	86.12	85.93	85.96	85.97	85.90	85.90	85.90
credit-g	75.95	76.45	76.46	76.41	76.33	76.52	76.30	76.20	76.45
cylinder-bands	84.43	77.52 •	78.28 •	78.37 •	77.67 •	77.46 •	76.94 •	77.44 •	76.83 •
diabetes	76.96	76.57	76.47	76.42	76.42	76.33	76.20	76.05	76.54
ecoli	85.43	81.67 •	82.59 •	82.50 •	81.52 •	82.71	80.17 •	79.97 •	80.56 •
energy-y1	66.03	58.58 •	62.93	62.28 •	58.18 •	57.69 •	57.56 •	57.93 •	59.32 •
energy–y2	54.56	50.05 •	52.00 •	51.92 •	50.29	50.05 •	49.74 •	49.80 •	50.10 •
glass	62.19	61.73	62.06	61.54	61.87	62.06	61.40	61.26	61.78
hayes-roth	81.00	71.00 •	71.44 •	71.38 •	69.94 •	71.13 •	71.06 •	71.19 •	70.75 •
heart-c	83.80	82.84	82.54	82.77	82.97	83.04	83.04	83.10	83.00
heart-h	83.85	84.09	83.85	83.86	84.32	84.29	84.50	84.30	84.36
heart-statlog	83.59	83.63	83.59	83.44	83.63	83.52	83.93	83.48	83.59
hepatitis	84.92	85.21	84.89	84.11	84.76	85.09	84.06	85.15	85.02
hypothyroid	94.36	93.56 •	93.65 •	93.65 •	93.61 •	93.58 •	93.52 •	93.58 •	93.62 •
ionosphere	93.44	91.85 •	92.02 •	92.00 •	91.99 •	91.88 •	91.85 •	91.74 •	91.74 •
iris	96.00	94.00	94.60	94.07	94.40	95.07	95.00	95.40	94.67
kr-vs-kp	94.62	91.64 •	94.75	94.14	92.30 •	93.26 •	94.14	93.51 •	91.03 •
labor	93.90	94.57	94.90	94.40	94.93	94.93	94.03	94.17	94.17
letter	83.28	77.80 •	78.94 •	78.79 •	78.06 •	78.65 •	78.70 •	78.87 •	77.64 •
lymph	85.67	85.46	85.52	85.46	85.39	85.59	85.46	85.91	85.92
mfeat-f	81.40	79.21 •	79.47 •	78.22 •	79.64 •	79.69 •	79.68 •	79.72 •	79.51 •
monks	99.45	82.23 •	99.87	99.85	80.16 •	74.64 •	74.64 •	99.75	81.89 •
movement-libras	81.22	76.08 •	76.11 •	72.81 •	76.11 •	76.06 •	75.97 •	76.06 •	76.03 •
mushroom	99.94	99.94	99.94	99.90	99.92	99.88	99.87	99.90	99.68
newthyroid	96.71	91.58 •	91.68 •	91.67 •	91.58 •	91.58 •	91.54 •	91.54 •	91.54 •
optdigits	96.49	95.67 •	95.80 •	95.69 •	95.72 •	95.65 •	95.82 •	95.85 •	95.77 •
page-blocks	93.57	93.22 •	93.36	93.35	93.32	93.23 •	93.28	92.87 •	93.28
pendigits	98.03	97.58 •	97.72 •	97.68 •	97.59 •	97.62 •	97.64 •	97.65 •	97.57 •
primary-tumor	48.38	47.87	48.05	47.84	47.87	47.96	47.70	47.61	47.72
qar-biodegradation	84.00	81.88 •	82.13 •	82.47 •	81.96 •	81.88 •	82.00 •	81.89 •	82.02 •
robot-24	91.40	89.66 •	89.89 •	88.47 •	89.70 •	89.71 •	89.64 •	89.68 •	89.66 •
segment	95.46	92.83 •	92.89 •	92.84 •	92.89 •	92.98 •	93.13 •	93.22 •	92.83 •
sick	98.14	97.74 •	97.97 •	97.95 •	97.59 •	97.63 •	98.01	97.84 •	97.52 •
sonar	78.38	79.91	80.25	78.38	80.20	80.20	80.30	80.91	79.92
soybean	94.45	93.32 •	93.35 •	93.44 •	93.38 •	93.28 •	93.26 •	93.32 •	93.31 •
spectrometer	54.20	48.12 •	48.31 •	45.73 •	48.18 •	48.10 •	48.21 •	48.16 •	47.97 •
splice	96.34	96.12	96.11	96.09	96.18	96.13	96.11	96.20	96.11
steel-plates-faults	92.32	90.07 •	91.92 •	95.77 ○	90.62 •	91.10 •	89.49 •	89.69 •	93.92 ○
texture	95.71	94.38 •	94.53 •	94.40 •	94.45 •	94.47 •	94.50 •	94.48 •	94.49 •
vehicle	73.88	71.65	71.93	70.39 •	71.58	71.85	71.83	71.70	71.64
vote	94.53	94.52	94.78	94.59	94.52	94.46	94.46	94.11	94.25
vowel	93.33	89.56 •	91.66 •	91.22 •	89.56 •	89.04 •	89.09 •	89.73 •	89.64 •
waveform	84.60	84.84	84.44	84.42	84.78	85.19	85.18	85.04	85.00
zoo	98.11	94.66	94.66	94.47	93.76	94.66	93.76	93.96	94.57
w/t/l	-	**0/25/31**	**0/29/27**	**1/27/28**	**0/27/29**	**0/26/30**	**0/28/28**	**0/27/29**	**1/26/29**

•, ○ :Statistically significant degradation/upgradation via a two-tailed t-test with 95% confidence level.

Table 4: Classification accuracy standard deviation comparisons on UCI data sets (%.)

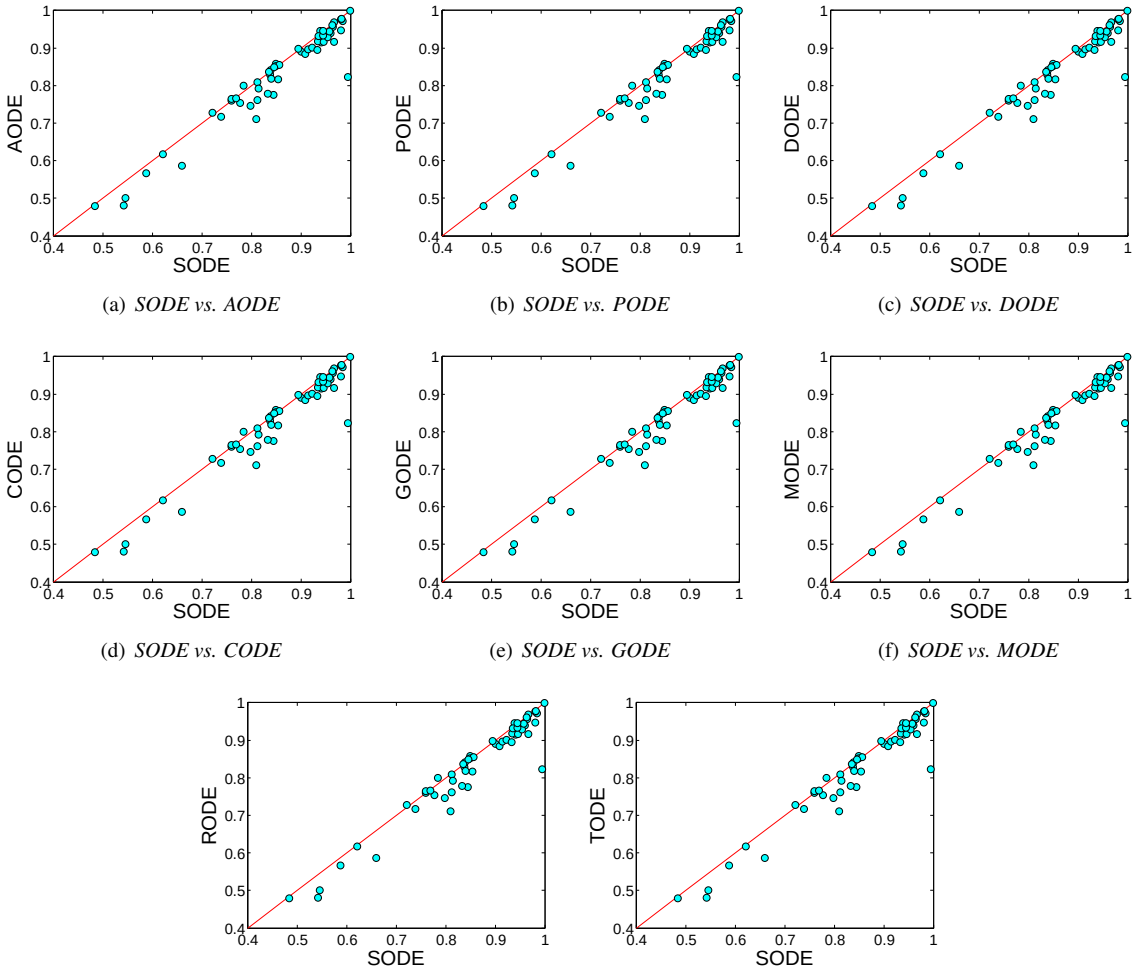
Data Set	**SODE**	AODE [40]	PODE [48]	DODE [49]	CODE [17]	GODE [52]	MODE [24]	RODE [34]	TODE [43]
anneal	1.05	1.66	1.61	1.61	1.59	1.56	1.58	1.51	1.59
anneal.ORIG	2.72	3.10	3.01	3.06	3.09	3.07	2.78	2.86	2.95
artificial-characters	0.97	1.52	1.46	1.47	1.42	1.48	1.44	1.47	1.47
audiology	6.37	6.42	6.42	6.46	6.48	6.42	6.67	6.59	6.54
autos	6.07	10.10	10.07	10.40	10.26	10.03	9.96	10.00	10.22
balance-scale	1.83	1.88	1.95	1.85	2.34	2.00	2.00	2.21	1.97
breast-cancer	6.45	7.01	7.08	7.05	6.88	7.33	7.11	7.13	7.11
breast-w	2.21	1.90	1.90	1.99	1.94	1.91	1.91	1.91	2.05
car	1.39	2.06	1.94	1.90	2.12	2.13	2.22	2.26	2.16
climate	1.70	2.45	2.73	2.47	2.49	2.92	2.92	2.80	2.46
colic	4.32	6.16	6.09	6.03	6.23	6.03	6.09	5.90	6.33
colic.ORIG	4.65	6.41	6.86	6.57	6.51	6.21	6.35	6.46	6.21
credit-a	3.55	3.72	3.61	3.60	3.73	3.58	3.57	3.57	3.76
credit-g	3.65	3.88	3.87	3.82	3.80	3.62	3.63	3.75	3.77
cylinder-bands	3.88	5.58	5.49	5.57	5.51	5.64	5.49	5.33	5.25
diabetes	4.27	4.53	4.35	4.78	4.58	4.64	4.64	4.70	4.52
ecoli	4.47	5.06	5.50	5.39	4.81	5.43	5.33	5.34	5.18
energy-y1	4.57	4.79	4.54	4.28	4.94	4.99	4.96	4.99	4.83
energy-y2	3.37	4.61	4.94	4.83	4.69	4.63	4.68	4.70	4.56
glass	9.45	9.69	9.20	9.25	9.43	9.46	9.38	9.24	9.68
hayes-roth	8.04	8.91	8.76	8.34	9.09	9.52	9.43	9.05	8.88
heart-c	6.04	7.03	6.90	7.01	6.91	6.93	6.83	6.91	7.00
heart-h	5.96	6.00	5.83	6.14	5.77	5.52	5.77	5.92	6.07
heart-statlog	5.37	5.32	5.60	5.73	5.67	5.80	5.84	5.90	5.89
hepatitis	6.25	9.36	9.57	9.97	9.65	9.63	10.05	9.49	9.51
hypothyroid	0.64	0.61	0.55	0.54	0.56	0.55	0.56	0.54	0.57
ionosphere	2.74	4.28	4.25	4.30	4.17	4.13	4.12	4.23	4.01
iris	4.76	5.88	5.82	5.91	5.50	5.16	5.14	4.80	5.53
kr-vs-kp	0.94	1.66	1.25	1.28	1.46	1.39	1.28	1.45	1.71
labor	9.26	9.72	9.61	9.34	9.13	9.13	10.18	9.90	9.48
letter	1.71	2.02	1.96	2.08	2.07	1.93	1.94	1.95	2.08
lymph	8.16	9.32	9.26	9.37	9.24	9.20	9.23	9.45	9.21
mfeat-f	2.24	2.45	2.49	2.56	2.39	2.35	2.33	2.29	2.40
monks	4.48	4.33	1.56	1.78	4.69	4.26	4.26	0.93	4.78
movement-libras	5.04	5.99	6.10	6.25	5.96	6.06	6.04	6.04	6.12
mushroom	0.19	0.19	0.19	0.24	0.21	0.24	0.25	0.23	0.42
newthyroid	4.31	5.02	5.04	5.00	5.02	5.02	5.04	5.04	5.04
optdigits	0.71	0.85	0.84	0.89	0.86	0.87	0.84	0.84	0.86
page-blocks	0.81	0.76	0.76	0.77	0.78	0.83	0.74	0.81	0.77
pendigits	0.35	0.41	0.39	0.39	0.40	0.40	0.39	0.40	0.42
primary-tumor	3.80	6.37	6.32	6.43	6.46	6.47	6.42	6.55	6.44
qar-biodegradation	3.31	3.88	4.00	3.74	3.91	3.85	3.88	3.92	3.88
robot-24	1.19	1.33	1.35	1.44	1.33	1.36	1.33	1.35	1.33
segment	1.08	1.40	1.41	1.55	1.42	1.44	1.49	1.45	1.43
sick	0.44	0.72	0.71	0.70	0.71	0.75	0.72	0.69	0.66
sonar	7.80	9.60	9.27	8.72	9.38	9.52	9.44	9.16	9.38
soybean	1.87	2.85	2.70	2.67	2.78	2.87	2.82	2.74	2.80
spectrometer	5.75	5.81	5.85	5.72	5.88	5.70	5.75	5.79	5.79
splice	0.72	1.00	1.03	1.04	0.98	0.99	1.01	0.98	1.01
steel-plates-faults	1.41	2.17	1.89	1.66	2.03	1.82	2.07	2.08	1.60
texture	0.76	1.01	0.95	1.00	0.94	0.97	0.96	0.96	0.96
vehicle	3.54	3.59	3.69	3.40	3.58	3.61	3.64	3.62	3.67
vote	3.17	3.19	3.24	3.22	3.19	3.17	3.17	3.35	3.28
vowel	2.18	3.06	2.95	2.96	3.09	3.21	3.21	3.15	3.13
waveform	3.41	3.07	3.20	3.24	3.24	3.11	3.14	3.15	3.08
zoo	4.22	6.38	6.38	6.64	6.43	6.38	6.43	6.46	6.50

Table 5: Experimental results on UCI data sets: Training Time (Sec).

Data Set	**SODE**	AODE [40]	PODE [48]	DODE [49]	CODE [17]	GODE [52]	MODE [24]	RODE [34]	TODE [43]
anneal	0.4517	0.0191	1.3995	1.3906	0.0488	0.0271	0.0155	0.1247	0.0649
anneal.ORIG	0.4411	0.0086	1.3709	1.3923	0.0264	0.0149	0.0145	0.1026	0.0407
artificial-characters	0.3969	0.0082	1.4227	1.0382	0.0178	0.0184	0.0082	0.2674	0.1673
audiology	1.2765	0.0104	4.3286	4.1895	0.0371	0.0168	0.0146	0.0649	0.0370
autos	0.0564	0.0026	0.1987	0.1680	0.0062	0.0038	0.0033	0.0185	0.0114
balance-scale	0.0029	0.0004	0.0323	0.0266	0.0006	0.0009	0.0003	0.0117	0.0062
breast-cancer	0.0034	0.0004	0.0312	0.0253	0.0010	0.0003	0.0005	0.0102	0.0051
breast-w	0.0086	0.0007	0.0620	0.0592	0.0021	0.0011	0.0008	0.0217	0.0058
car	0.0175	0.0008	0.1620	0.1465	0.0022	0.0017	0.0010	0.0457	0.0102
climate	0.0320	0.0023	0.1848	0.1084	0.0063	0.0039	0.0035	0.0324	0.0115
colic	0.0254	0.0015	0.1057	0.0866	0.0045	0.0025	0.0023	0.0287	0.0115
colic.ORIG	0.0352	0.0024	0.1532	0.1187	0.0068	0.0038	0.0035	0.0337	0.0144
credit-a	0.0226	0.0012	0.1111	0.0935	0.0036	0.0025	0.0018	0.0354	0.0145
credit-g	0.0596	0.0030	0.2921	0.2105	0.0088	0.0054	0.0046	0.0693	0.0334
cylinder-bands	0.1243	0.0168	0.5992	0.4144	0.0471	0.0286	0.0275	0.0881	0.0441
diabetes	0.0082	0.0005	0.0499	0.0469	0.0017	0.0012	0.0009	0.0224	0.0117
ecoli	0.0097	0.0006	0.0433	0.0422	0.0012	0.0007	0.0007	0.0110	0.0047
energy-y1	0.1230	0.0016	0.3593	0.3124	0.0035	0.0026	0.0021	0.0343	0.0243
energy-y2	0.1273	0.0017	0.3583	0.3201	0.0037	0.0028	0.0025	0.0368	0.0219
glass	0.0088	0.0003	0.0409	0.0350	0.0013	0.0010	0.0006	0.0080	0.0058
hayes-roth	0.0007	0.0001	0.0176	0.0134	0.0004	0.0003	0.0001	0.0023	0.0022
heart-c	0.0157	0.0004	0.0683	0.0609	0.0017	0.0011	0.0009	0.0148	0.0068
heart-h	0.0150	0.0004	0.0687	0.0591	0.0014	0.0010	0.0008	0.0135	0.0055
heart-statlog	0.0073	0.0005	0.0467	0.0357	0.0020	0.0011	0.0009	0.0133	0.0066
hepatitis	0.0071	0.0005	0.0428	0.0350	0.0019	0.0010	0.0007	0.0066	0.0041
hypothyroid	0.7534	0.0196	2.5204	2.7864	0.0576	0.0370	0.0334	0.3290	0.2331
ionosphere	0.0604	0.0044	0.2553	0.1868	0.0131	0.0074	0.0073	0.0406	0.0164
iris	0.0006	0.0002	0.0150	0.0133	0.0004	0.0004	0.0001	0.0021	0.0012
kr-vs-kp	0.4937	0.0239	2.8490	2.3393	0.0722	0.0439	0.0406	0.3749	0.1199
labor	0.0022	0.0003	0.0216	0.0175	0.0010	0.0005	0.0006	0.0014	0.0017
letter	1.9621	0.0146	4.7750	4.2716	0.0325	0.0228	0.0210	0.2412	0.1815
lymph	0.0109	0.0010	0.0499	0.0465	0.0017	0.0010	0.0010	0.0060	0.0043
mfeat-f	8.6214	0.2714	22.7282	20.6407	0.5124	0.4233	0.4076	0.7125	0.6794
monks	0.0036	0.0006	0.0412	0.0472	0.0007	0.0009	0.0008	0.0141	0.0051
movement-libras	2.9361	0.1259	8.3806	7.9377	0.1739	0.1713	0.1875	0.2980	0.2613
mushroom	0.1088	0.0052	0.3676	0.7140	0.0158	0.0099	0.0086	0.1351	0.0185
newthyroid	0.0014	0.0002	0.0191	0.0170	0.0006	0.0003	0.0002	0.0041	0.0015
optdigits	16.8904	0.3355	45.5101	42.6542	0.7521	0.4563	0.5530	1.9393	1.1122
page-blocks	0.2148	0.0045	0.7298	0.8794	0.0140	0.0092	0.0080	0.1657	0.0644
pendigits	2.0822	0.0288	5.5798	5.7605	0.0743	0.0534	0.0456	0.7943	0.2723
primary-tumor	0.1170	0.0010	0.3533	0.3477	0.0023	0.0018	0.0015	0.0296	0.0159
qar-biodegradation	0.2697	0.0166	1.1033	0.7953	0.0446	0.0291	0.0279	0.1582	0.0881
robot	0.9376	0.0291	3.1681	2.5756	0.0749	0.0527	0.0490	0.5288	0.1945
segment	0.4158	0.0086	1.0964	1.3094	0.0207	0.0141	0.0130	0.1387	0.0523
sick	0.3983	0.0195	1.8619	2.1304	0.0572	0.0349	0.0323	0.3222	0.0852
sonar	0.1101	0.0101	0.4463	0.3249	0.0341	0.0172	0.0206	0.0426	0.0336
soybean	0.9237	0.0066	2.5657	2.5088	0.0170	0.0112	0.0105	0.1041	0.0347
spectrometer	16.3370	0.2567	44.9328	45.2391	0.4701	0.4129	0.4001	0.6494	0.6812
splice	2.4584	0.0887	6.7361	6.8725	0.2148	0.1487	0.1439	0.8177	0.3503
steel-plates-faults	0.3269	0.0196	1.5014	1.6842	0.0448	0.0324	0.0306	0.2166	0.0671
texture	6.7507	0.1081	18.6764	17.3391	0.2136	0.1524	0.1568	0.9611	0.4256
vehicle	0.0834	0.0033	0.2842	0.2259	0.0071	0.0051	0.0046	0.0502	0.0301
vote	0.0134	0.0009	0.0774	0.0691	0.0021	0.0016	0.0013	0.0242	0.0058
vowel	0.1331	0.0024	0.4033	0.3212	0.0048	0.0038	0.0036	0.0487	0.0247
waveform	0.3623	0.0261	1.1789	0.9782	0.0415	0.0296	0.0276	0.1362	0.0841
zoo	0.0099	0.0002	0.0550	0.0419	0.0009	0.0007	0.0005	0.0038	0.0022

Table 6: Experimental Results on UCI data sets: Testing Time (Sec).

Data Set	**SODE**	AODE [40]	PODE [48]	DODE [49]	CODE [17]	GODE [52]	MODE [24]	RODE [34]	TODE [43]
anneal	0.0068	0.0082	0.0074	0.0058	0.0052	0.0050	0.0060	0.0051	0.0054
anneal.ORIG	0.0039	0.0052	0.0050	0.0050	0.0050	0.0048	0.0049	0.0049	0.0047
artificial-characters	0.0045	0.0042	0.0042	0.0047	0.0050	0.0046	0.0042	0.0045	0.0043
audiology	0.0155	0.0166	0.0153	0.0157	0.0201	0.0158	0.0151	0.0174	0.0188
autos	0.0001	0.0009	0.0008	0.0007	0.0009	0.0005	0.0009	0.0007	0.0007
balance-scale	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0003
breast-cancer	0.0002	0.0001	0.0001	0.0001	0.0001	0.0002	0.0001	0.0001	0.0001
breast-w	0.0005	0.0001	0.0003	0.0002	0.0001	0.0001	0.0002	0.0001	0.0001
car	0.0003	0.0003	0.0003	0.0003	0.0003	0.0001	0.0002	0.0005	0.0004
climate	0.0009	0.0003	0.0005	0.0004	0.0006	0.0004	0.0005	0.0003	0.0004
colic	0.0005	0.0003	0.0005	0.0003	0.0001	0.0003	0.0003	0.0003	0.0002
colic.ORIG	0.0005	0.0002	0.0005	0.0005	0.0004	0.0004	0.0005	0.0004	0.0004
credit-a	0.0007	0.0004	0.0004	0.0003	0.0001	0.0001	0.0003	0.0005	0.0001
credit-g	0.0006	0.0008	0.0007	0.0007	0.0006	0.0009	0.0009	0.0004	0.0007
cylinder-bands	0.0012	0.0016	0.0012	0.0014	0.0017	0.0015	0.0020	0.0016	0.0016
diabetes	0.0004	0.0002	0.0001	0.0002	0.0002	0.0002	0.0001	0.0001	0.0004
ecoli	0.0002	0.0001	0.0001	0.0002	0.0001	0.0001	0.0001	0.0001	0.0002
energy-y1	0.0016	0.0014	0.0018	0.0018	0.0015	0.0016	0.0016	0.0016	0.0014
energy-y2	0.0016	0.0018	0.0017	0.0018	0.0017	0.0016	0.0015	0.0027	0.0014
glass	0.0002	0.0003	0.0003	0.0001	0.0003	0.0001	0.0002	0.0003	0.0001
hayes-roth	0.0001	0.0001	0.0001	0.0001	0.0001	0.0002	0.0001	0.0001	0.0001
heart-c	0.0005	0.0001	0.0002	0.0007	0.0002	0.0003	0.0003	0.0003	0.0001
heart-h	0.0003	0.0005	0.0001	0.0003	0.0005	0.0002	0.0001	0.0005	0.0006
heart-statlog	0.0002	0.0001	0.0002	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
hepatitis	0.0002	0.0003	0.0003	0.0002	0.0001	0.0001	0.0002	0.0001	0.0002
hypothyroid	0.0141	0.0085	0.0086	0.0085	0.0082	0.0084	0.0081	0.0081	0.0083
ionosphere	0.0012	0.0006	0.0006	0.0007	0.0008	0.0007	0.0007	0.0008	0.0008
iris	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
kr-vs-kp	0.0092	0.0059	0.0059	0.0060	0.0059	0.0058	0.0054	0.0059	0.0058
labor	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
letter	0.0369	0.0199	0.0193	0.0208	0.0222	0.0201	0.0202	0.0196	0.0198
lymph	0.0002	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
mfeat-f	0.1374	0.1308	0.1515	0.1304	0.1320	0.1296	0.1300	0.1356	0.1487
monks	0.0002	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0002	0.0001
movement-libras	0.0565	0.0403	0.0431	0.0505	0.0403	0.0567	0.0565	0.0560	0.0652
mushroom	0.0020	0.0012	0.0012	0.0013	0.0014	0.0011	0.0012	0.0012	0.0012
newthyroid	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
optdigits	0.2215	0.2050	0.2274	0.2163	0.2140	0.1826	0.2055	0.2153	0.2111
page-blocks	0.0043	0.0022	0.0030	0.0034	0.0084	0.0020	0.0024	0.0024	0.0022
pendigits	0.0209	0.0209	0.0218	0.0214	0.0216	0.0210	0.0208	0.0208	0.0213
primary-tumor	0.0014	0.0013	0.0014	0.0014	0.0013	0.0014	0.0014	0.0012	0.0014
qar-biodegradation	0.0029	0.0026	0.0031	0.0026	0.0026	0.0026	0.0028	0.0027	0.0028
robot	0.0101	0.0093	0.0116	0.0105	0.0097	0.0090	0.0093	0.0090	0.0090
segment	0.0080	0.0043	0.0059	0.0053	0.0045	0.0081	0.0042	0.0044	0.0041
sick	0.0037	0.0045	0.0063	0.0055	0.0042	0.0044	0.0048	0.0043	0.0046
sonar	0.0021	0.0013	0.0013	0.0016	0.0017	0.0014	0.0013	0.0013	0.0015
soybean	0.0107	0.0105	0.0099	0.0129	0.0105	0.0104	0.0102	0.0102	0.0098
spectrometer	0.3180	0.3475	0.3470	0.3787	0.3717	0.3580	0.3736	0.3778	0.3451
splice	0.0252	0.0242	0.0235	0.0248	0.0230	0.0241	0.0285	0.0236	0.0235
steel-plates-faults	0.0063	0.0031	0.0032	0.0036	0.0030	0.0032	0.0032	0.0030	0.0031
texture	0.0772	0.0706	0.0773	0.0703	0.0704	0.1226	0.0788	0.0724	0.0765
vehicle	0.0014	0.0008	0.0009	0.0009	0.0007	0.0009	0.0009	0.0007	0.0010
vote	0.0006	0.0001	0.0003	0.0003	0.0001	0.0002	0.0001	0.0002	0.0003
vowel	0.0028	0.0014	0.0015	0.0015	0.0016	0.0015	0.0014	0.0018	0.0014
waveform	0.0035	0.0035	0.0037	0.0041	0.0037	0.0035	0.0037	0.0037	0.0038
zoo	0.0003	0.0001	0.0005	0.0002	0.0002	0.0001	0.0001	0.0001	0.0001



(a) *SODE vs. AODE* (b) *SODE vs. PODE* (c) *SODE vs. DODE*

(d) *SODE vs. CODE* (e) *SODE vs. GODE* (f) *SODE vs. MODE*

(g) *SODE vs. RODE* (h) *SODE vs. TODE*

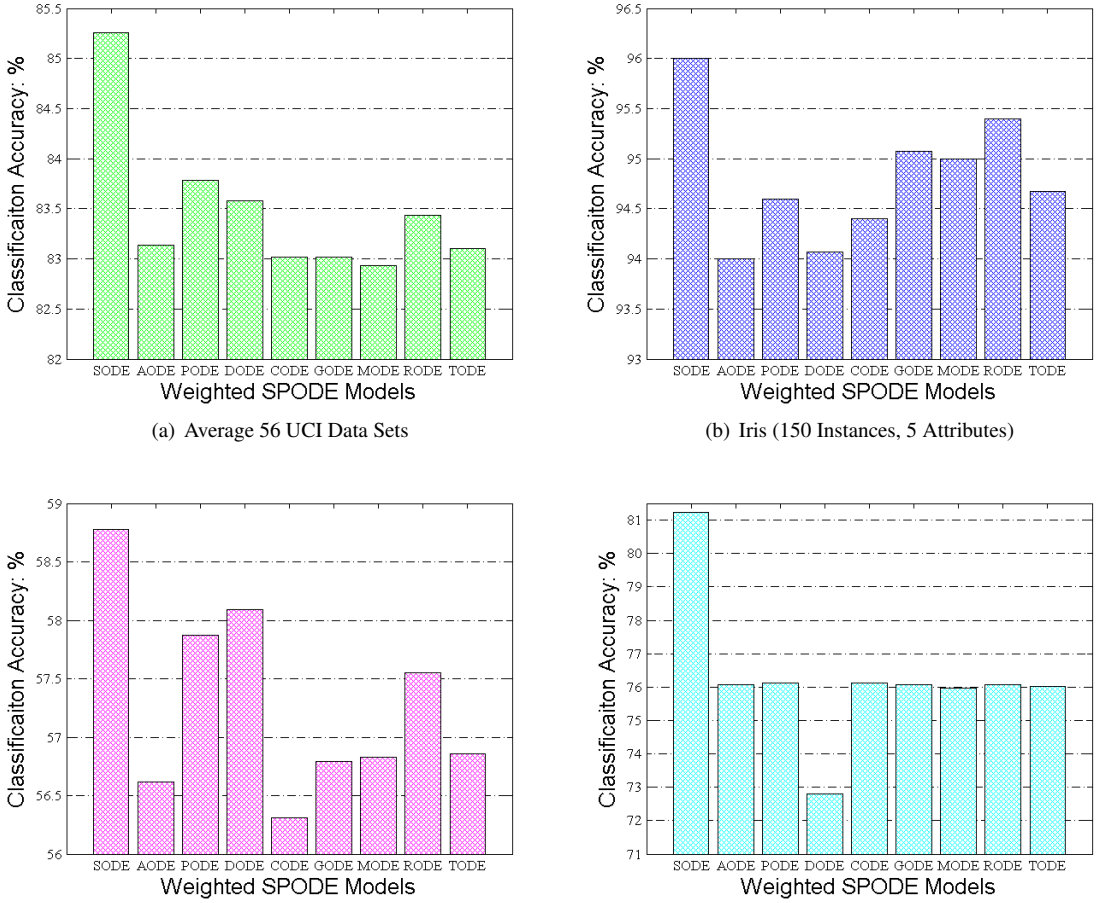
Figure 5: Head-to-head comparison between SODE *vs.* rival weighted SPODE algorithms on 56 UCI data sets. Each data point in figure represents classification accuracy on one data set. The $x-$axis denotes SODE's accuracy and the $y-$axis represents the rival method's accuracy. A data point below $y = x$ diagonal line indicates that SODE outperforms the rival method.

5.3. Comparisons on UCI benchmark data sets

We first report the classification accuracies on the fifty-six benchmark data sets from UCI repository[1], which includes data from a wide range of domains (the data characteristics are briefly described in Table 2).

In our experiments, we compare the effectiveness of SODE with AODE [40], PODE [48], DODE [49], CODE [17], GODE [52], MODE [24], RODE [34] and TODE [43]. For all benchmark data sets, we comparatively study the performance of the proposed SODE *w.r.t.* other baselines, and report the results in Figure 5, where data points below the $x = y$ diagonal line are data sets on which SODE achieves better results than the rival algorithm.

[1]https://archive.ics.uci.edu/ml/datasets.html



(a) Average 56 UCI Data Sets

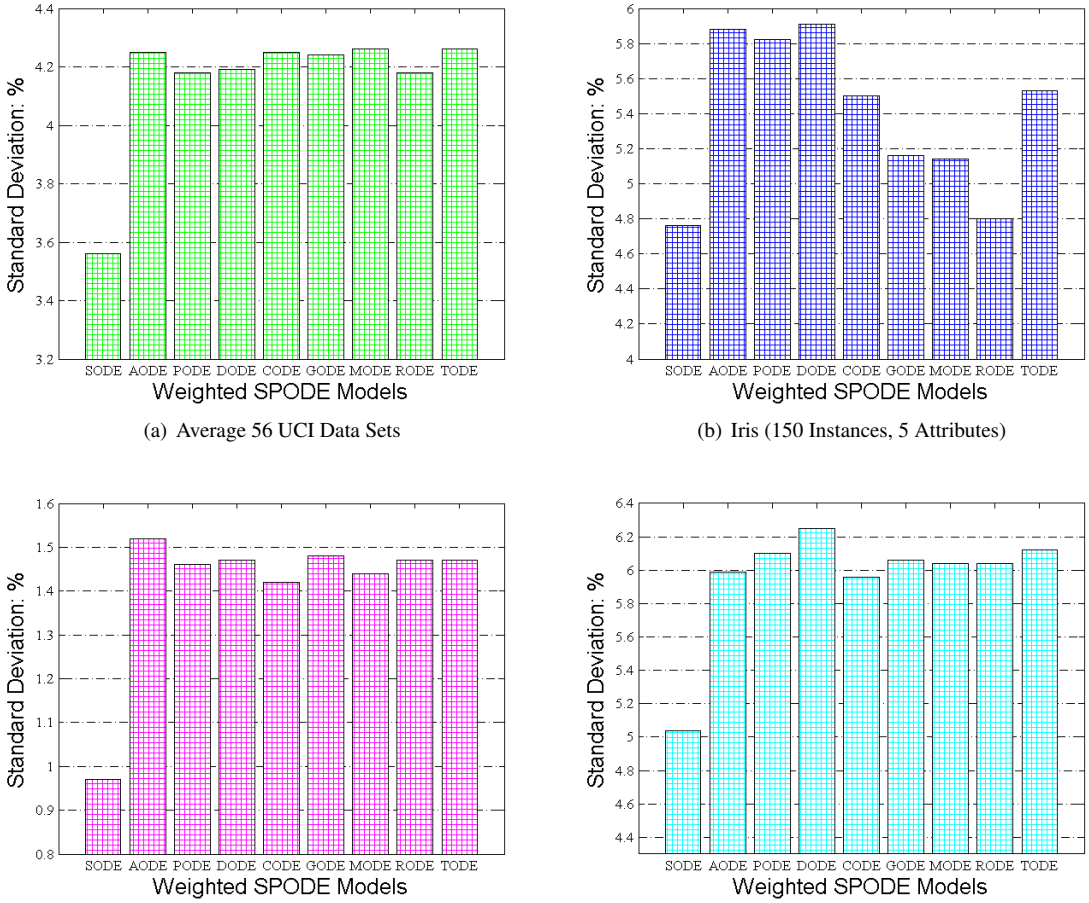
(b) Iris (150 Instances, 5 Attributes)

(c) Artificial-Characters (10218 Instances, 8 Attributes)

(d) Movement-Libras (360 Instances, 91 Attributes)

Figure 6: Classification accuracy on (a) all UCI benchmark data sets, and (b) "Iris" data with 150 instances and 5 attributes, (c) "Artificial-Characters" data with 10218 instances and 8 attributes, and (d) "Movement-Libras" data with 360 instances and 91 attributes respectively.

In Table 3, we also report the detailed accuracy of all methods on the 56 benchmark data sets reported in Figure 6(a). For each row (*i.e.* a data set) in Table 3, a field marked with • and ◦ mean that compared to the method showing in the corresponding column, SODE's classification accuracy is statistically and significantly better (upgradation) or worse (degradation), respectively, using two-tailed t-test with 95% confidence level. The entry $w/t/l$ at the bottom of table means that the algorithm in the corresponding column wins in w data sets, ties in t data sets, and loses in l data sets on the 56 benchmark data sets, compared to SODE. In addition to the classification accuracy, standard deviation, *i.e.,* the square root of the variance, describes the component of error that results from random factors, such as random variation in the training data and random processor in the learning algorithm, and therefore measures the robustness of the algorithm (*i.e.,* how sensitive an algorithm facing the data changes). Table 4 shows the detailed standard deviations *w.r.t.* each data set, and the average of the stand deviations are shown in Figure 7(a).



(a) Average 56 UCI Data Sets

(b) Iris (150 Instances, 5 Attributes)

(c) Artificial-Characters (10218 Instances, 8 Attributes)

(d) Movement-Libras (360 Instances, 91 Attributes)

Figure 7: Standard deviation of classification accuracy on (a) all UCI benchmark data sets, and (b) "Iris" data with 150 instances and 5 attributes, (c) "Artificial-Characters" data with 10218 instances and 8 attributes, and (d) "Movement-Libras" data with 360 instances and 91 attributes respectively.

Overall, the results can be summarized as follows:

1. Figure 5 shows that majority data points fall below the diagonal line $x = y$, which indicates SODE performs better than other baseline weighted SPODE models.

2. SODE greatly outperforms the classical AODE model (31 wins and 0 losses) and the gain ratio weighted GODE (30 wins and 0 losses). The average accuracy on 56 data sets for SODE (85.26%) is higher than both AODE (83.14%) and GODE (83.02%).

3. SODE significantly outperforms both CODE and RODE with 29 wins and 0 losses. Although the average accuracy of RODE (83.43%) is superior to CODE (83.02%), they are both inferior to the proposed SODE.

4. Maximum a posteriori linear mixture of discriminative distributions MAPLMD/DODE shows the superi-

ority in average accuracy (83.58%) compared to two weighted baselines, including MODE (82.93%) and TODE (83.11%). However, by self-adaptively adjusting the attribute weight for each SPODE, SODE consistently outperforms TODE (29 wins and 1 losses), MODE (28 wins and 0 losses), and MAPLMD/DODE (28 wins and 1 losses).

5. Existing empirical studies have suggested that the Bayesian model averaging of maximum a posteriori linear mixture of generative distribution MAPLMG/PODE is one of the most effective of approaches [48, 49]. Compared to our proposed SODE, it loses 27 on data sets, and its average accuracy 83.78% is also inferior to SODE.

6. From the standard deviation perspective, it is observed that, SODE achieves the best performance in reducing the standard deviation of the accuracy compared to other alternative approaches. This is mainly attributed to the self-adaptive adjusting strategy used in SODE, so the ensemble of the SPODEs can have less variance in their prediction and therefore be more stable.

In order to further demonstrate the algorithm performance in different data environments (*e.g.,* number of instances or attributes), we first report the experimental results on "Iris" data set with a small number of instances (150 instances) and 5 attributes in Figure 6(b), followed by two other special data sets "Artificial-Characters" (10218 Instances, 8 Attributes) with relatively a large number of instances and "Movement-Libras" (360 Instances, 91 Attributes) for large number of attributes in Figures 6(c) and 6(d), respectively. The standard deviation estimation could also be found in Figures 7(b), 7(c), and 7(d). As expected, SODE also demonstrates the best classification performance with high accuracy and low standard deviation.

5.4. Image Retrieval Learning Task

In image classification, an image is classified into different categories according to its visual content. An important application of image classification is image retrieval: searching through an image data set to obtain (or retrieve) those images with particular (or user provided) visual content. For example, finding pictures containing a car.

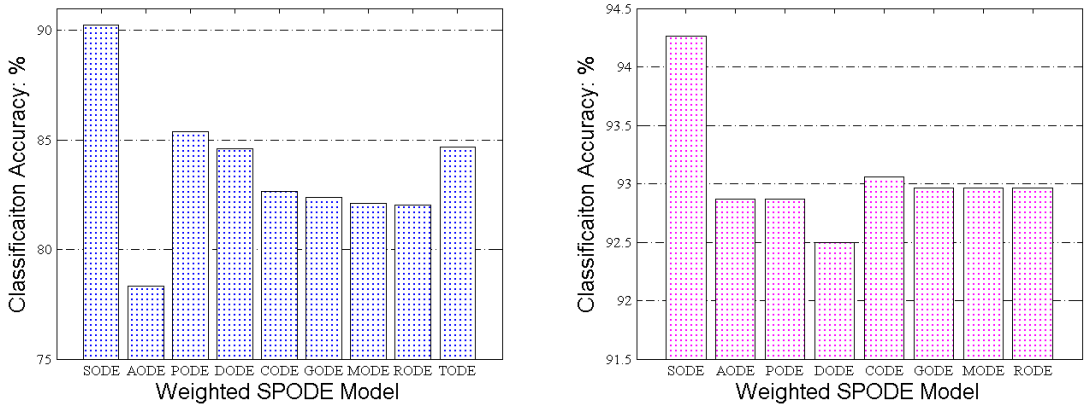
In this part of experiment, we report SODE's performance for content-based image retrieval task. In our experiments, we obtain the original color images from Corel data set [27]. For each image, four sets of visual features [20] are extracted, including color histogram , color histogram layout, color moments, and co-occurrence texture. We choose the color histogram approach in the HSV color space as color features. The HSV color space is divided into 32 subspaces (32 colors with 8 ranges of H and 4 ranges of S). After that, the value in each dimension in a color histogram of an image is the density of each color in the entire image, which yields 32-dimensional color histogram features. For the color histogram layout, each image is divided into 4 sub-images (one horizontal split and one vertical split), in which 4×2 color histogram for each subimage is computed. In this case, we can obtain another 32-dimensional features. In addition, the color moment feature has 9 dimensions, in which one (mean, or standard deviation, or skewness) for each of H, S, and V in HSV color space. At last,

Figure 8: Example images used in the experiment from the COREL image categorization database. The first three rows show images in category "Cats", including "Lion" (The first row), "Tiger" (The second row) and "Leopard" (The third row), and the last three rows represent the image examples form other categories. Examples show that image retrieval is challenging and many non-cat images are also visually similar to the ones showing in the first three rows. The similar image data generation can also be found in previous work [46].

for texture feature, images are converted to 16 gray-scale images, then co-occurrence in 4 directions is computed (horizontal, vertical, and two diagonal directions). The corresponding 16 texture feature values are: one for each direction, second angular moment, contrast, inverse difference moment, and entropy.

In our experiment, we use category "Cats" as the positive class, which consists of "Tiger", "Lion" and "Leopard", to form a binary learning problem (each subcategory has 100 images). To obtain negative classes, we selected 300 images randomly from the remaining classes. Some sample images from the benchmark data sets are shown in Figure 8.

Figure 9(a) shows the accuracy of SODE and the baselines. We can see that all weighted SPODE classifiers achieve a higher accuracy than unweighted AODE, which equally combines SPODEs for prediction. This is mainly attributed to the fact that attributes are playing different roles in each individual learning tasks, and therefore should be differentiated during the learning (and the classification) process. For CODE, GODE, MODE and RODE, they have similar performance gain but are all inferior to the TODE and DODE. On the other hand, MAPLMG/PODE which is one of the most effective approaches in literature, has shown the best performance among the baseline weighted SPODE models. In fact, MAPLMG (Maximum a Posteriori Linear Mixture of Generative Distributions) employs a maximum a posterior principle to determine weight values for SPODEs. This resembles to the fitness principles employed in SODE which intends to find the optimal generative parameters maximally approximate to the given training data. Although MAPLMG employs unconstrained maximization to find parameters, SODE employs immune procedures go generate diversified parameters and self-adaptively



(a) Image (b) Text

Figure 9: Experimental results on (a) Image data set and (b) Text data set, respectively: Accuracy %.

search for the optimal parameter settings for classification.

5.5. Text Categorization Task

Our text categorization data set contains documents of free text business descriptions of Brazilian companies, categorized into a subset of 9 categories cataloged in a table called national classification of economic activities[2]. The original texts were pre-processed as follows to build our data set: 1) Initially, prepositions of the texts are removed and only letters remain for processing; 2) Secondly, words are transformed to their canonical form; and 3) each document is represented by a vector with 857 features (552 of which are binary), where the weight of each word is its frequency in the document. The categories are equally distributed, with 120 instances in each of nine categories (*i.e.,* 1080 documents in total). Because the main purpose for this paper is to design a good weighting model for ensembling SPODEs, and our analysis on the 56 benchmark data sets has already demonstrated the performance of the proposed SODE, we use the text and image learning tasks to demonstrate the generality of the SODE for different applications. Moreover, the SPODE model (*e.g.,* AODE) has already been improved for the highly scalable attribute problem in [10]. Also, SPODE models can be trained incrementally [5]. When facing huge word histograms, a hashing correlated feature approach in [4] has been proposed to rank the features. Similarly, some other feature selection/extraction methods can also be united with the SPODEs for classification. Because the paper is not primarily targeting ultra-high dimensional data, we did not report results in this regard.

The text categorization data set is very sparse (99.22% of the matrix is filled with zeros). To alleviate the data sparsity issue, dimensionality reduction is applied to retain the top 200 words with the highest information gain score [11]. As a result, each document/instance is represented by a 200-dimensional feature vector.

The results in the Figure 9(a) show that MAPLMD/DODE's performance is inferior to unweighted AODE. This suggests that this type of attribute weighting approaches, maximizing a posteriori linear mixture of discrim-

[2]http://archive.ics.uci.edu/ml/datasets/CNAE-9

inative distributions, do not work well for text data, possibly because of the high data dimensionality. Meanwhile MAPLMG/PODE, which is competitive to GODE, MODE, and RODE, is inferior to the CODE which uses correlation-based weighting model. By employing self-adaptive weighting strategy, SODE demonstrates significant performance gain, compared to other attribute weighted and unweighted baselines, especially the MAPLMG/PODE (the most effective approach in literature).

5.6. Detailed Algorithm Performance Studies

5.6.1. Time Complexity Analysis

Training Time Complexity. The time complexity of SODE mainly includes the following two parts: (1) evaluation of SODE, and (2) updating of the weight values.

Prior to the evaluation of SODE model, SODE needs to build n single SPODE classifier from data set $\mathcal{D}^a$ with N_a instances, which will take $O(N_a \cdot n^2)$, where n is the number of attribute (each individual SPODE needs to scan the whole training set and builds prior probabilities for all classes and conditional probabilities for all n attributes). For the weight population $\mathcal{W}$ in each generation, the calculation of affinity function for each weight individual $\mathbf{w} \in \mathcal{W}$ is similar to testing all SPODE classifiers on a test set $\mathcal{D}^b$ with N_b instances, which will take $O(c \cdot N_b \cdot n^2 \cdot L)$, where L is the size of the weight populations (*i.e.,* the number of weight vectors). The rest four operations (*e.g.,* selection, clone, mutation, and update) are all based on weight vectors. The corresponding time complexity is $O(L \cdot logn)$. Assume the average number of evolution generations is M, the total time complexity $\mathcal{U}$ is given by Eq. (12).

$$\mathcal{U} = O(N_a \cdot n^2) + M \times [O(c \cdot N_b \cdot n^2 \cdot L) + O(L \cdot logn)] \tag{12}$$

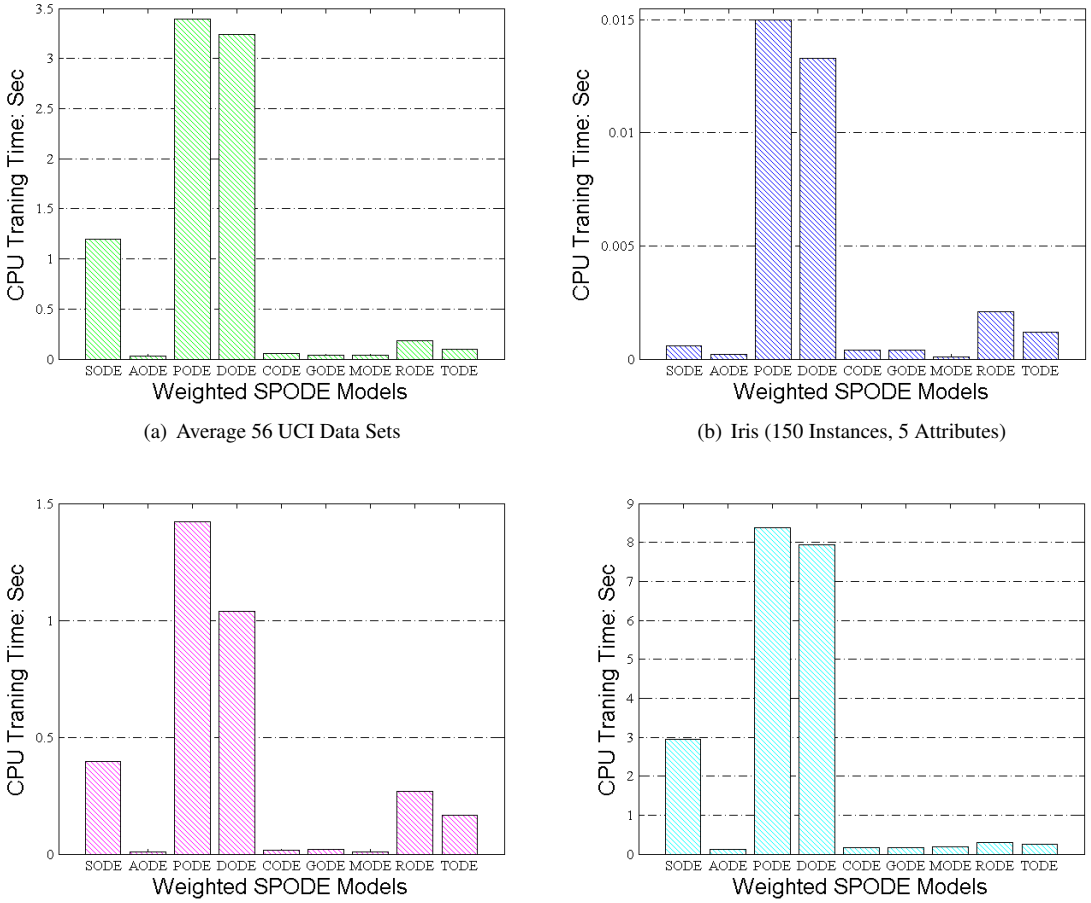
Because $N_a + N_b = N$, where N is the total number of training instances, Eq. (12) can be rewritten as

$$\begin{aligned} \mathcal{U} =& O[(N - N_b) \cdot n^2] + O(c \cdot N_b \cdot n^2 \cdot L \cdot M) + O(L \cdot logn \cdot M) \\ \leq& O(N \cdot n^2) + O(c \cdot N_b \cdot n^2 \cdot L \cdot M) + O(L \cdot logn \cdot M) \\ \leq& O(c \cdot N \cdot n^2 \cdot L \cdot M) + O(L \cdot logn \cdot M) \leq O(c \cdot N \cdot n^2 \cdot L \cdot M) \end{aligned} \tag{13}$$

Eq. (13) shows that the total time complexity of SODE is mainly bounded by four important factors: (1) the total number of training instances N; (2) the number of the attributes n; (3) the size of weight pollution L; and (4) the average number of evolution generations M. In our experiments, we use a threshold T to automatically determine the termination process by following the principle that if the result gap obtained from two consecutive

Table 7: Training Time Complexity.

AODE [40]	PODE [48]	DODE [49]	CODE [17]	GODE [52]	MODE [24]	RODE [34]	TODE [43]
$O(Nn^2)$	O(cNn^2+ckNn)	O(cNn^2+ckNn)	$O(Nn^2)$	$O(Nn^2)$	$O(Nn^2)$	$O(cN^2n^2)$	$O(Nn^3)$



(a) Average 56 UCI Data Sets

(b) Iris (150 Instances, 5 Attributes)

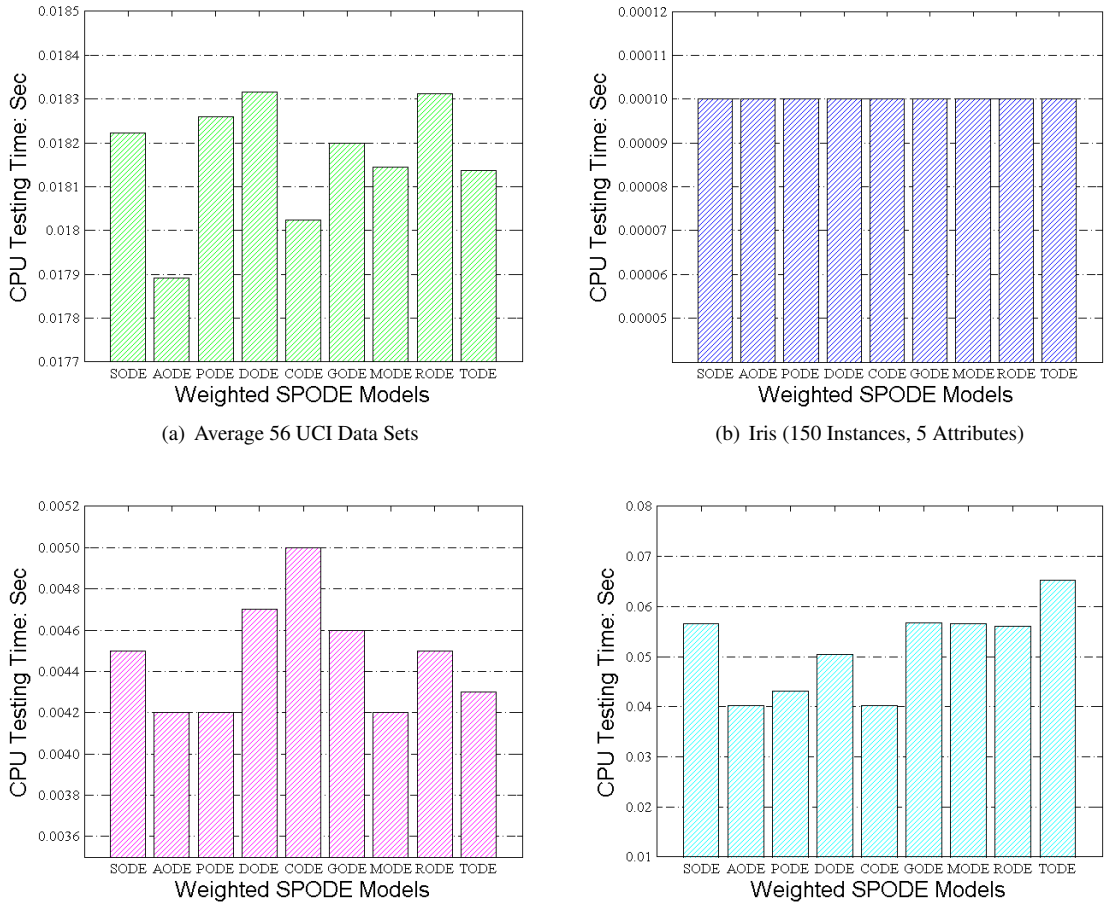
(c) Artificial-Characters (10218 Instances, 8 Attributes)

(d) Movement-Libras (360 Instances, 91 Attributes)

Figure 10: Training time on (a) the UCI benchmark data sets and, (b) "Iris" data with 150 instances and 5 attributes, (c) "Artificial-Characters" data with 10218 instances and 8 attributes, and (d) "Movement-Libras" data with 360 instances and 91 attributes, respectively.

iterations is less than T, the algorithm will terminate. This process will further reduce the number of iterations and save the computational cost.

In Table 7, we summarize the time complexity of other baselines for comparisons. During the training process, GODE and MODE both need to calculate the gain ratio and mutual information as attribute weights from the whole training data with $O(Nn^2)$ complexity. In the subsequent SPODE model training, the total training overhead is $O(Nn^2 + Nn^2) \leq O(Nn^2)$. For CODE, it needs to build a CFS model, with $O(N((n^2 - n)/2))$ computational complexity. Therefore, the corresponding overall complexity is $O(N((n^2 - n)/2) + Nn^2) \leq O(Nn^2)$. For RODE, its Relief-F model is subject to the complexity $O(cN^2n)$, therefore the total complexity is $O(cN^2n + Nn^2) \leq O(cN^2n^2)$. TODE requires to build a C4.5 tree with $O(Nn^3)$ complexity, consisting of building a tree $O(Nnlogn)$ and subtree replacement and pruning $O(n(logn)^2$. As a result, the training time complexity of TODE will be $O(Nn^3 + Nn^2) \leq O(Nn^3)$. For DODE with MAPLMD, it first trains a supervised posterior probability model



(a) Average 56 UCI Data Sets

(b) Iris (150 Instances, 5 Attributes)

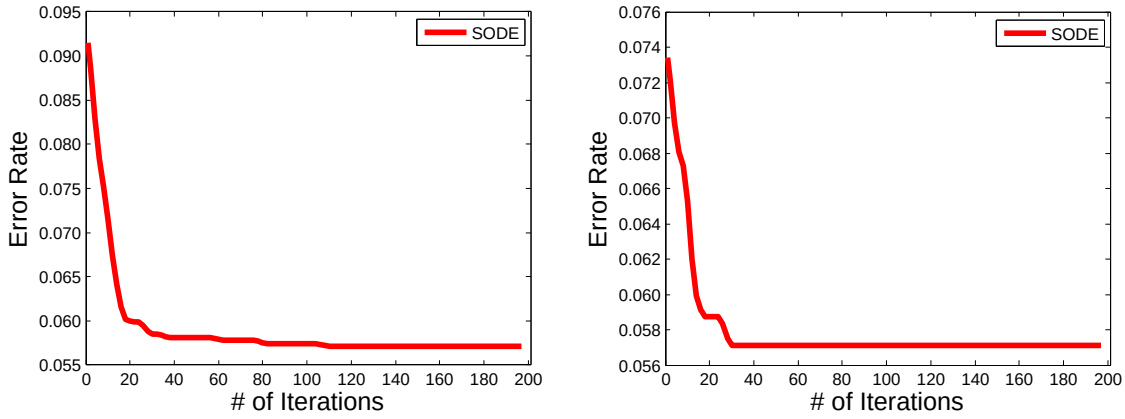
(c) Artificial-Characters (10218 Instances, 8 Attributes)

(d) Movement-Libras (360 Instances, 91 Attributes)

Figure 11: Testing time on (a) the UCI benchmark data sets, and (b) "Iris" data with 150 instances and 5 attributes, (c) "Artificial-Characters" data with 10218 instances and 8 attributes, and (d) "Movement-Libras" data with 360 instances and 91 attributes respectively.

with $O(cNn^2)$ complexity, followed by an EM algorithm with a large fixed number K that bounds the number of iterations which attributes to $O(cKNn)$ complexity. Therefore, DODE will cost $O(cNn^2 + ckNn)$. In GODE with MAPLMG, it also first trains a posterior probability model as in DODE with $O(cNn^2)$ complexity. The difference is that GODE uses BFGS minimization algorithm with k iteration convergence. In our experiments, the parameters in PODE and DODE are set the same as the ones used in [48] and [49].

Testing Time Complexity. For the testing time complexity, all algorithms are subject to $O(cn^2)$ costs, because after the weights are calculated from the training process, the testing can be directly carried out using obtained weight values. Notice that, according to our time complexity analysis, CODE, GODE, and MODE have the same time complexity. The similar observation can be found between PODE and DODE. However, although these methods have the same asymptotic complexity, their actual CPU runtime may vary significantly. Accordingly, in the following subsection, we further carry out CPU runtime analysis.



(a) Kr-vs-kp Data (b) Soybean Data

Figure 12: Convergence curves of SODE for (a) "Kr-vs-kp", and (b) "Soybean" data, respectively.

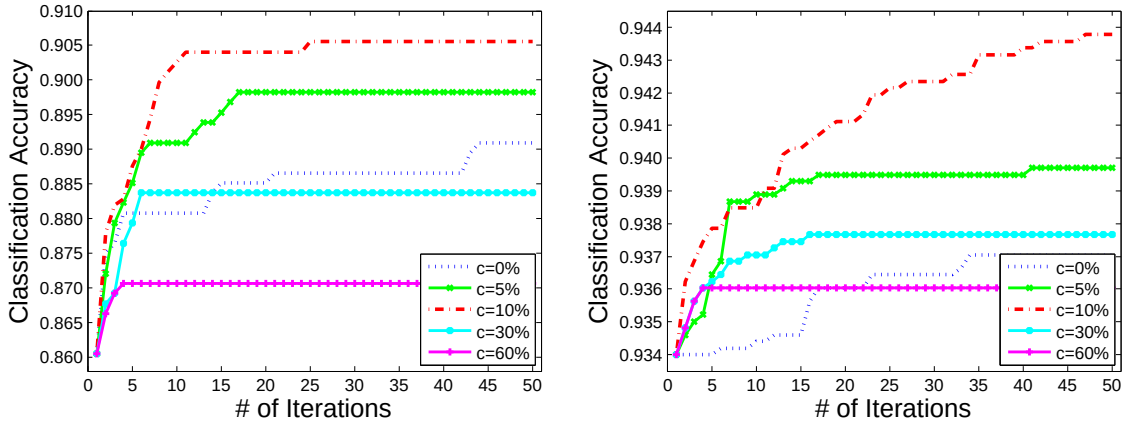
5.6.2. CPU Runtime Analysis

In Figure 10(a), we report the average CPU training time on 56 UCI benchmark data sets. The detailed results for each method, with respect to each data set, are reported in Table 5. Overall, the results show that AODE, which does not have any weighting scheme, demonstrates the best efficiency. However, existing studies [48, 22] have also validated that AODE can not achieve better classification performance than weighted SPODE models. On the other hand, an empirical study [49] has suggested that Bayesian model averaging of "Maximum a Posteriori Linear Mixture of Generative Distributions" (MAPLMG) is one of the most effective approaches. Our results show that the proposed SODE is not only more accurate than MAPLMG/PODE, but is also more efficient in terms of CPU runtime. It is worth noting that, from the result on a general data set "Iris" with 150 instances and 5 attributes in Figure 10(b), SODE can obtain a competitive efficiency with AODE. Because the efficiency of SPODE family models may vary on data sets with a large number of instances or attributes, we further report the CPU runtime on a number of data sets with different characteristics. Figures 10(c) and 10(d) show the CPU training time results on "Artificial-Characters" data with 10218 instances and 8 attributes and "Movement-Libras" data with 360 instances and 91 attributes, respectively. SODE consistently demonstrates better runtime than PODE and DODE.

Table 6 also lists the details of CPU runtime for testing on 56 UCI benchmark data sets, with Figure 11(a) reporting the average CPU testing time on all UCI data sets. In addition, we also report the result of the general "Iris" data set in Figure 11(b), Figure 11(c) for "Artificial-Characters" data set, and Figure 11(d) for "Movement-Libras" data set, respectively. Because weight values are only calculated during the training process, all algorithms have similar/comparable CPU testing runtime.

5.6.3. Convergence and Learning Curves

In order to investigate the convergence of the SODE algorithm, we validate the relationship between the number of iterations and the classification error rate on the "Kr-vs-kp", and "Soybean" data sets, and report



(a) Energy-y1 Data

(b) Robot-24 Data

Figure 13: The results with respect to different clone factors on (a) "Energy-y1", and (b) "Robot-24" data sets, respectively.

the results in Figure 12. Each point in the curves corresponds to the mean error rate from the 10-fold cross validation under the iteration with the current optimal attribute weight values. Overall SODE demonstrates a fast convergence and a lower classification error rate than other algorithms, which implies that the impact of the parameter maximum iteration *MaxGen* in SODE is insignificant. For detailed investigations, we observe the algorithm performance on "Kr-vs-kp" data set, which has 37 dimensions and 3,196 instances. Some previous studies [25, 23] have discovered strong attribute dependencies in this data set. Our result in Figure 12(a) shows that SODE achieves a low classification error 0.06 after 20th iteration, which demonstrates the convergence speed of SODE. Moreover, there is no noticeable change after the 20th iteration, although SODE still does not converge yet. In this case, the parameter T in SODE can help early terminate the algorithm, which will further reduce the number of iterations and reduce the computational cost. Similar runtime improvement can also be observed from other data sets.

5.6.4. Effectiveness of SODE Clone Strategy

The clone strategy plays an important role in SODE, as it helps generate diversified weight candidates. In order to study the impact of the clone factor parameter c used in SODE, we report the learning curves with respect to different c values. In Figure 13, we report the convergence curves with c values varying form 0%, 5%, 10%, 30%, to 60% on "Energy-y1" Figure 13(a), and "Robot-24" Figure 13(b), respectively. Notice that, when c value is set to zero, there is no clone strategy involved in this version, *i.e.,* a general evolutionary algorithm. The results from Figure 13 show that clone factor c has a significant impact on the algorithm performance, and SODE without clone scheme almost has the worst effectiveness (as shown in Figure 13(b)). On the other hand, having too many clones also deteriorates the classification accuracy, mainly because clones severely reduce the diversity in the weight population. For both Figures 13 (a) and (b), a clone factor $c = 60\%$ results in the worst performance. By contrast, SODE with clone factor c being $5\% \sim 10\%$ is significantly superior to the one without

clone. This observation demonstrates that SODE with effective clone strategy outperforms the ones using a general evolutionary strategy.

6. Conclusions and Further Work

In this paper, we proposed to use immune principle to self-adaptively determine the optimal attribute weight values for SPODEs. Different from existing attribute weighting approaches, which typically assess/rank attribute based on predefined measures, our proposed method, namely SODE, intends to combine attribute weighting and the derivation of the learning model into a unified objective function. To this end, we use immunity theory to search optimal weight values for SPODE, so the determined weight values can automatically adapt to the underlying data distributions to ensure the algorithm performance. Experiments and comparisons on 56 UCI data sets and validations in image retrieval and text categorization tasks show that SODE outperforms state-of-the-art attribute weighted SPODE approaches in terms of classification accuracy and standard deviation. Experiments and analysis on time complexity and CPU runtime performance further demonstrate SODE provides effective trade-off between runtime efficiency and accuracy effectiveness.

However, the application tasks (text category or image retrieval) are just demos in this paper. When handing a huge number of dimensions in current day image data (*e.g.,* 1M dimensional FV [2] and 4096 dimensional ConvNets [9] which are not categorical), SPODEs family will pre-discretize the attribute with continuous variables into categorical. Although the error of the probabilistic techniques could also be reduced by application of alternative discretization techniques [39], we can apply the technologies proposed in GAODE and HAODE [14] to directly deal with the continuous variables.

Acknowledgments

We thank Dr. Jesus Cerquides and Prof. Geoffrey I. Webb for their kind help on the implementation details of MAPLMD/DODE and MAPLMG/PODE. This work was partially supported by the Australian Research Council Discovery Projects under Grant Nos. DP140100545 and DP140102206.

References

[1] Aguiar, P., Xing, E. P., Figueiredo, M., Smith, N. A., & Martins, A. (2011). An augmented lagrangian approach to constrained map inference. In *Proceedings of the 28th International Conference on Machine Learning* ICML' 11 (pp. 169–176). New York, NY, USA.

[2] Akata, Z., Perronnin, F., Harchaoui, Z., & Schmid, C. (2014). Good Practice in Large-Scale Learning for Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*, 507–520.

[3] Babu, G., & Murty, M. (1994). Clustering with evolution strategies. *Pattern Recognition*, *27*, 321 – 329.

[4] Bai, B., Weston, J., Grangier, D., Collobert, R., Sadamasa, K., Qi, Y., Chapelle, O., & Weinberger, K. (2010). Learning to rank with (a lot of) word features. *Inf. Retr.*, *13*, 291–314.

[5] Bouckaert, R. R. (2006). Voting massive collections of bayesian network classifiers for data streams. In *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence* AI'06 (pp. 243–252).

[6] de Castro, L., & Von Zuben, F. (2002). Learning and optimization using the clonal selection principle. *Evolutionary Computation, IEEE Transactions on*, *6*, 239–251.

[7] Castro, L. N. D., & Timmis, J. (2002). Artificial immune systems: A novel paradigm to pattern recognition. In *Artificial Neural Networks in Pattern Recognition* (pp. 67–84). Springer Verlag, University of Paisley, UK.

[8] Cerquides, J., & de Mántaras, R. L. (2005). Robust bayesian linear classifier ensembles. In *Proceedings of the 16th European Conference on Machine Learning* ECML' 05 (pp. 72–83). Berlin, Heidelberg.

[9] Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the 25th British Machine Vision Conference* BMVC'14.

[10] Chen, S., Martinez, A., & Webb, G. (2014). Highly scalable attribute selection for aode. In *Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 86–97).

[11] Ciarelli, P. M., & Oliveira, E. (2009). Agglomeration and elimination of terms for dimensionality reduction. In *Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications* ISDA '09 (pp. 547–552). Washington, DC, USA.

[12] Domingos, P. (2000). Bayesian averaging of classifiers and the overfitting problem. In *Proceedings of the Seventeenth International Conference on Machine Learning* ICML (pp. 223–230). San Francisco, CA, USA.

[13] Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc.

[14] Flores, M. J., Gámez, J. A., Martínez, A. M., & Puerta, J. M. (2009). Gaode and haode: Two proposals based on aode to deal with continuous variables. In *Proceedings of the 26th Annual International Conference on Machine Learning* ICML'09 (pp. 313–320).

[15] Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*, 131–163.

[16] Hall, M. (2007). A decision tree-based attribute weighting filter for naive bayes. *Knowledge-Based Systems*, *20*, 120–126.

[17] Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning* ICML '00 (pp. 359–366). San Francisco, CA, USA.

[18] Hernández-González, J., Inza, I. n., & Lozano, J. A. (2013). Learning bayesian network classifiers from label proportions. *Pattern Recogn.*, *46*, 3425–3440.

[19] Hoeting, J. A., Madigan, D., Raftery, A. E., & Volinsky, C. T. (1999). Bayesian model averaging: A tutorial. *STATISTICAL SCIENCE*, *14*, 382–417.

[20] Hong, Z., Mei, X., Prokhorov, D., & Tao, D. (2013). Tracking via robust multi-task multi-view joint sparse representation. In *Proceedings of the 2013 IEEE International Conference on Computer Vision* ICCV'13 (pp. 649–656). Sydney, Australia.

[21] Huang, K., Liu, X., Li, X., Liang, J., & He, S. (2013). An improved artificial immune system for seeking the pareto front of land-use allocation problem in large areas. *International Journal of Geographical Information Science*, *27*, 922–946.

[22] Jiang, L., & Zhang, H. (2006). Weightily averaged one-dependence estimators. In *Proceedings of the 9th Pacific Rim International Conference on Artificial Intelligence* PRICAI'06 (pp. 970–974). Berlin, Heidelberg: Springer-Verlag.

[23] Jiang, L., Zhang, H., & Cai, Z. (2009). A novel bayes model: Hidden navie bayes. *IEEE Transactions on Knowledge and Data Engineering*, *21*, 1361–1371.

[24] Jiang, L., Zhang, H., Cai, Z., & Wang, D. (2012). Weighted average of one-dependence estimators. *Journal of Experimental and Theoretical Artificial Intelligence*, *24*, 219–230.

[25] Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers:a decision-tree hybrid. In *Proceedings of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD'96 (pp. 202–207). New York.

[26] Langseth, H., & Nielsen, T. D. (2006). Classification using hierarchical naive bayes models. *Mach. Learn.*, *63*, 135–159.

[27] Li, J., & Wang, J. Z. (2008). Real-time computerized annotation of pictures. *IEEE Trans. Pattern Anal. Mach. Intell.*, *30*, 985–1002.

[28] de Mello Honorio, L., Leite da Silva, A., & Barbosa, D. (2012). A cluster and gradient-based artificial immune system applied in optimization scenarios. *Evolutionary Computation, IEEE Transactions on*, *16*, 301–318.

[29] Mokhtari, A., & Ribeiro, A. (2014). Res: Regularized stochastic bfgs algorithm. *Signal Processing, IEEE Transactions on*, *62*, 6089–6104.

[30] Monteith, K., Carroll, J., Seppi, K., & Martinez, T. (2011). Turning bayesian model averaging into bayesian model combination. In *Neural Networks (IJCNN), The 2011 International Joint Conference on* (pp. 2657–2663).

[31] Park, T., & Ryu, K. R. (2010). A dual-population genetic algorithm for adaptive diversity control. *Evolutionary Computation, IEEE Transactions on*, *14*, 865–884.

[32] Polat, K., Gne, S., & Tosun, S. (2006). Diagnosis of heart disease using artificial immune recognition system and fuzzy weighted pre-processing. *Pattern Recognition*, *39*, 2186 – 2193.

[33] Polat, K., Gne, S., & Tosun, S. (2011). Corrigendum to diagnosis of heart disease using artificial immune recognition system and fuzzy weighted pre-processing. *Pattern Recognition*, *44*, 1327.

[34] Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning*, *53*, 23–69.

[35] Rodin, V., Benzinou, A., Guillaud, A., Ballet, P., Harrouet, F., Tisseau, J., & Bihan, J. L. (2004). An immune oriented multi-agent system for biological image processing. *Pattern Recognition*, *37*, 631 – 645.

[36] Rokach, L. (2008). Genetic algorithm-based feature set partitioning for classification problems. *Pattern Recognition*, *41*, 1676 – 1700.

[37] Sahami, M. (1996). Learning limited dependence bayesian classifiers. In *Proceedings of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD '96 (pp. 335–338). New York.

[38] Triguero, I., Garca, S., & Herrera, F. (2011). Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification. *Pattern Recognition*, *44*, 901 – 916.

[39] Webb, G. I., Boughton, J., & Wang, Z. (2002). Averaged one-dependence estimators: Preliminary results. In *Proceedings of the Australasian Data Mining Workshop* (pp. 65–73).

[40] Webb, G. I., Boughton, J. R., & Wang, Z. (2005). Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, *58*, 5–24.

[41] Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems (2nd ed.). Morgan Kaufmann Publishers.

[42] Woldemariam, K. M., & Yen, G. G. (2010). Vaccine-enhanced artificial immune system for multimodal function optimization. *Trans. Sys. Man Cyber. Part B*, *40*, 218–228.

[43] Wu, J., & Cai, Z. (2011). Learning averaged one-dependence estimators by attribute weighting. *Journal of Information and Computational Science*, *8*, 1063–1073.

[44] Wu, J., Cai, Z., Zeng, S., & Zhu, X. (2013). Artificial immune system for attribute weighted naive bayes classification. In *Proceedings of the International Joint Conference on Neural Networks* IJCNN'13 (pp. 798–805). Dallas, TX, USA.

[45] Wu, J., Cai, Z., & Zhu, X. (2013). Self-adaptive probability estimation for naive bayes classification. In *Proceedings of the International Joint Conference on Neural Networks* IJCNN'13 (pp. 2303–2310). Dallas, TX, USA.

[46] Wu, J., Hong, Z., Pan, S., Zhu, X., Zhang, C., & Cai, Z. (2014). Multi-graph learning with positive and unlabeled bags. In *Proceedings of SIAM International Conference on Data Mining* SDM'14 (pp. 217–225). Philadelphia, Pennsylvania, USA.

[47] Yang, Y., Korb, K., Ting, K. M., & Webb, G. I. (2005). Ensemble selection for superparent-one-dependence estimators. In *Proceedings of the 18th Australian Joint Conference on Advances in Artificial Intelligence* AI'05 (pp. 102–112). Sydney, Australia.

[48] Yang, Y., Webb, G., Cerquides, J., Korb, K., Boughton, J., & Ting, K. M. (2006). To select or to weigh: a comparative study of model selection and model weighing for spode ensembles. In *ECML* (pp. 533–544).

[49] Yang, Y., Webb, G. I., Cerquides, J., Korb, K. B., Boughton, J., & Ting, K. M. (2007). To select or to weigh: A comparative study of linear combination schemes for superparent-one-dependence estimators. *IEEE Trans. on Knowl. and Data Eng.*, *19*, 1652–1665.

[50] Yuan, J., Zhang, L., Zhao, C., Li, Z., & Zhang, Y. (2012). An improved self-organization antibody network for pattern recognition and its performance study. *Pattern Recognition*, *321*, 96–103.

[51] Zaidi, N. A., Cerquides, J., Carman, M. J., & Webb, G. I. (2013). Alleviating naive bayes attribute independence assumption by attribute weighting. *Journal of Machine Learning Research*, *14*, 1947–1988.

[52] Zhang, H., & Sheng, S. (2004). Learning weighted naive bayes with accurate ranking. In *Proceedings of the Fourth IEEE International Conference on Data Mining* ICDM'04 (pp. 567–570). Washington, DC, USA.

[53] Zheng, F., & Webb, G. I. (2006). Efficient lazy elimination for averaged one-dependence estimators. In *Proceedings of the 23rd International Conference on Machine Learning* ICML '06 (pp. 1113–1120). Pittsburgh, Pennsylvania.

[54] Zheng, F., Webb, G. I., Suraweera, P., & Zhu, L. (2012). Subsumption resolution: an efficient and effective technique for semi-naive bayesian learning. *Mach. Learn.*, *87*, 93–125.

[55] Zheng, J., Chen, Y., & Zhang, W. (2010). A survey of artificial immune applications. *Artif. Intell. Rev.*, *34*, 19–34.

[56] Zhong, Y., & Zhang, L. (2013). Sub-pixel mapping based on artificial immune systems for remote sensing imagery. *Pattern Recognition*, *46*, 2902 – 2926.