Neurocomputing 191 (2016) 363–373

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

KNN-based dynamic query-driven sample rescaling strategy for class imbalance learning

Jun Hu [a], Yang Li [a], Wu-Xia Yan [a], Jing-Yu Yang [a], Hong-Bin Shen [b], Dong-Jun Yu [a,*]

[a] School of Computer Science and Engineering, Nanjing University of Science and Technology, Xiaolingwei 200, Nanjing 210094, China
[b] Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Dongchuan Road 800, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 11 June 2015
Received in revised form
21 October 2015
Accepted 27 January 2016
Communicated by D. Wang
Available online 10 February 2016

Keywords:
Imbalanced learning
Sample rescaling
Dynamic query-driven sample rescaling
Classifier ensemble
Protein–nucleotide binding residues prediction

ABSTRACT

The class imbalance phenomenon is pervasive in bioinformatics prediction problems in which the number of majority samples is significantly larger than that of minority samples. Relieving the severity of class imbalance has been demonstrated to be a promising route for enhancing the prediction performance of a statistical machine learning-based predictor under an imbalanced learning scenario. In this study, we propose a novel *d*ynamic *q*uery-*d*riven *s*ample *r*escaling (DQD-SR) strategy for addressing class imbalance. Unlike the traditional sample rescaling technique, which often yields a fixed balanced dataset, the proposed DQD-SR dynamically generates a query-driven balanced dataset based on KNN algorithm. A prediction model trained on a *t*raditional *s*ample *r*escaling (T-SR)-derived balanced dataset will partially learn the global knowledge buried in the original dataset, whereas a prediction model trained on DQD-SR will reflect the query-specific local knowledge between a query sample and its correlated neighbors in the original dataset. Thus, we developed an ensemble scheme to integrate the T-SR-based model and the DQD-SR-based model to further improve the overall prediction performance. To demonstrate the efficacy of the proposed method, we performed stringent cross-validation and independent validation tests on benchmark datasets concerning protein–nucleotide binding residues prediction, which is a typical imbalanced learning problem in bioinformatics. Computer experimental results show that the proposed method achieves high prediction performance and outperforms existing sequence-based protein–nucleotide binding residues predictors. We also implemented a predictor called TargetNUCs, which is freely available for academic use at http://csbio.njust.edu.cn/bioinf/TargetNUCs.

1. Introduction

The class imbalance phenomenon, in which the number of majority samples is significantly larger than that of minority samples, is pervasive in the fields of bioinformatics and machine learning [1–5] and has aroused wide concern from academia and government funding agencies [6,7]. Previous studies have demonstrated that directly applying statistical learning algorithms, which assume samples in different classes are balanced, to class imbalance problems will often leads to poor performance [7–10]. One of the underlying possible reasons is that the under-represented minority class cannot draw the same amount of attention as the majority class by statistical learning algorithms, which in turn results in very specific classification rules or missing rules for the minority class [3,11]. Hence, developing effective computational methods for imbalanced learning has become a popular topic in recent machine learning and bioinformatics research [1,8,12].

Over the past decades, numerous computational methods for imbalanced learning problems have emerged. In [6], He and Garcia presented a comprehensive and extensive survey on the progress made in imbalanced learning. These imbalanced learning methods can be roughly grouped into three categories: data-level, algorithm-level, and ensemble-level methods [3]. Data-level methods include a variety of sample rescaling techniques that try to rectify the degree of skewness by manipulating the distribution of samples in different classes. Under-sampling and over-sampling techniques, such as *r*andom *u*nder-*s*ampling (RUS) [6], *r*andom *o*ver-*s*ampling (ROS) [6], *s*ynthetic *m*inority *o*ver-sampling *te*chnique (SMOTE) [13], *ada*ptive *syn*thetic sampling (ADASYN) [14], *s*upervised *o*ver-sampling (SOS) [15], are commonly used sample rescaling strategies. Generally, under-sampling decreases the size of the majority class by removing samples from the original dataset, whereas over-sampling alters the size of the minority class by replicating the randomly selected samples or synthesizes

* Corresponding author. Tel.: +86 25 84316190; fax: +86 25 84315960.
E-mail addresses: njyudj@126.com, njyudj@njust.edu.cn (D.-J. Yu).

http://dx.doi.org/10.1016/j.neucom.2016.01.043
0925-2312/

new samples across the original sample distribution [8]. To date, no theoretical justification or evidence has shown that under-sampling prevails over over-sampling or vise versa; hence, both are widely used in imbalanced learning problems.

Algorithm-level methods address imbalanced learning problems by directly changing the training mechanism of the traditional machine learning algorithms with the goal of achieving better performance on the minority class [3]. Representative methods include one-class learning [16] and cost-sensitive learning algorithms [17]. Algorithm-level methods target the imbalanced learning problem by requiring specific treatments for different types of machine learning algorithms, which hinders their generalizability in many fields because the best learning algorithm in most cases cannot be chosen in advance and modifying their training mechanism is a difficult task [3].

Ensemble learning [18] addresses imbalanced learning problems by integrating multiple machine-learning-based classifiers, such as SMOTEBoost [19], DataBoost-IM [9], and AdaBoost.NC [20,21]. Ensemble learning algorithms have been verified to be able to combine learning abilities from many individual classifiers and improve overall accuracy [22,23].

Although many studies have succeeded in learning from imbalanced data, in real applications, existing approaches usually train a fixed global prediction model based on different imbalanced learning strategies [8,15,24,25]; for any future query inputs, the trained fixed global model is used to perform prediction without considering the differences in characteristics between the query inputs. Specific treatments for different query inputs have been shown to be able to avoid the risk of over-optimization associated with static models and enhance learning performance [26].

Therefore, we propose a novel *d*ynamic *q*uery-*d*riven *s*ample *r*escaling (DQD-SR) strategy for addressing class imbalance at the data level. Unlike the traditional sample rescaling technique used at the data level, through which a fixed global prediction model is trained for all future query inputs, the proposed DQD-SR dynamically generates a query-driven balanced dataset based on the *k*-*n*earest neighbor algorithm (KNN) [27] and then trains a query-specific prediction model on the balanced dataset for a given query input. More concretely, the proposed DQD-SR balances the dataset by choosing the minority and majority parts from the minority and majority samples, respectively, according to the KNN-based local distribution information correlated to each query input.

Generally, the balanced dataset generated by the *t*raditional *s*ample *r*escaling (T-SR) technique, such as RUS or ROS, partially reflects the global distribution information of samples regardless of the characteristics of query inputs, whereas the balanced dataset generated with DQD-SR reflects the local distribution information of samples correlated to a given query input. Accordingly, a prediction model trained on a T-SR-based balanced dataset will partially learn the global knowledge buried in the original dataset, whereas a prediction model trained on DQD-SR will reflect the query-specific local knowledge between a query sample and its correlated neighbors in the original dataset. Thus, we developed a scheme to ensemble the T-SR-based model and the DQD-SR-based model and thereby improve overall prediction performance.

To demonstrate the efficacy of the proposed DQD-SR and the ensemble scheme for imbalanced learning, we applied the abovementioned methods to solve the protein–nucleotide binding residue prediction problem, a typical imbalanced learning problem in bioinformatics.

Protein–nucleotide interactions are indispensable in virtually all biological processes [28–30], and the prediction of protein–nucleotide binding residues using automated computational approaches has become a popular topic in bioinformatics [8,15,24,25,31,32]. In the post-genome era, with rapid developments in sequencing technology and concerted genome projects, large volumes of protein sequences have been accumulated without being functionally annotated; thus, designing computational methods for protein–nucleotide binding residue prediction solely from sequence information would be particularly useful. Many efforts have been made to achieve protein–nucleotide binding residue prediction over the past decade [31–33]. Nevertheless, to the best of our knowledge, most existing predictors are fixed static prediction models that do not carefully consider the characteristics of dynamic query inputs. Therefore, this paper presents a new dynamic protein–nucleotide binding residue predictor called TargetNUCs, which ensembles a DQD-SR-based model and a T-SR-based model with the proposed ensemble scheme. It should be noted that we utilized RUS as an example of T-SR and SVM as the base model engine for implementing TargetNUCs.

We compared the TargetNUCs with other popular sequence-based protein–nucleotide binding residue predictors on three benchmark datasets. The experimental results show that TargetNUCs achieves high prediction performance and outperforms most existing protein–nucleotide binding residues predictors. We also implemented a web server of TargetNUCs, which is freely available for academic use available at http://csbio.njust.edu.cn/bioinf/TargetNUCs.

2. Proposed DQD-SR and ensemble scheme

As described in Section 1, the dataset generated with T-SR partially reflects the global distribution information of samples regardless of the characteristics of query inputs. We believe that the local distribution information correlated to a query input is also important for predicting its class label, especially in an imbalanced learning scenario. Motivated by this hypothesis, we thus propose the new DQD-SR algorithm to dynamically construct a query-specific balanced dataset that partially reflects the local distribution information correlated to the query input.

Furthermore, a decision-level ensemble scheme is proposed to effectively utilize the global knowledge of the model trained on a T-SR-based balanced dataset and the local knowledge of the model trained on a DQD-SR-based balanced dataset.

2.1. Proposed DQD-SR

Let $S^{tr}=\{(\mathbf{x}_i^{tr},y_i^{tr})\}_{i=1}^{N}$ be the original training dataset, where $\mathbf{x}_i^{tr}\in R^m$ and $y_i^{tr}\in\{+1,-1\}$ are the m-dimensional feature vector and the class label, respectively, of the ith sample. Here, $+1$ and -1 represent the positive and negative classes, respectively. Thus, for each query input $\mathbf{x}_j^{tst}\in R^m$, the corresponding local training dataset LS_j^{tr} constructed from S^{tr} can be represented by a so-called index vector $\mathbf{a}_j=[a_{j,1},a_{j,2},\cdots,a_{j,i},\cdots,a_{j,N}]^T$, $a_{j,i}\in\{0,1\}$, where $a_{j,i}=1$ indicates that the ith sample in S^{tr} is selected as the neighbor of the query input $\mathbf{x}_j^{tst}$ and $a_{j,i}=0$ indicates that the ith sample is not selected.

In this study, we utilize the KNN algorithm, which has been successfully used in many under-sampling strategies [2], to construct query-specific local training dataset, i.e., LS_j^{tr}, as follows:

First, we represent the original dataset as follows:

$$S^{tr}=S_{pos}^{tr}\cup S_{neg}^{tr} \tag{1}$$

where S_{pos}^{tr} and S_{neg}^{tr} represent the sets of positive and negative samples, respectively.

Then, we use the KNN algorithm to obtain num_{pos} nearest neighbors of the query input $\mathbf{x}_j^{tst}$ from S_{pos}^{tr} and the corresponding index vector, denoted as $\mathbf{a}_j^{pos}=[a_{j,1}^{pos},\cdots,a_{j,i}^{pos},\cdots,a_{j,|S_{pos}^{tr}|}^{pos}]^T$:

$$\mathbf{a}_j^{pos}=KNNSelection(\mathbf{x}_j^{tst},S_{pos}^{tr},num_{pos}) \tag{2}$$

where $a_{j,i}^{pos} \in \{0,1\}$ denotes whether a sample in S_{pos}^{tr} is selected as the neighbor of the query input and $|\cdot|$ denotes the cardinality of a set.

Algorithm 1. DQD-SR algorithm.

Input: $S^{tr} = \{(\mathbf{x}_i^{tr}, y_i^{tr})\}_{i=1}^{N}$ – the original training dataset;
$\mathbf{x}_j^{tst}$ – the query input;
num_{pos} – the number of selected positive samples;
num_{neg} – the number of selected negative samples.

Output: LS_j^{tr} – the rescaled training dataset for the query input $\mathbf{x}_j^{tst}$.

Procedure:

1 Divide S^{tr} into two parts (i.e., S_{pos}^{tr} and S_{neg}^{tr}) based on each label (y_i^{tr}) information:
$(S_{pos}^{tr}, S_{neg}^{tr}) = DivideDataset(S^{tr})$

2 Use KNN algorithm to select a positive selected vector ($\mathbf{a}_j^{pos}$) and a negative selected vector ($\mathbf{a}_j^{neg}$) from S_{pos}^{tr} and S_{neg}^{tr}, respectively:
$\mathbf{a}_j^{pos} = KNNSelection(\mathbf{x}_j^{tst}, S_{pos}^{tr}, num_{pos})$,
$\mathbf{a}_j^{neg} = KNNSelection(\mathbf{x}_j^{tst}, S_{neg}^{tr}, num_{neg})$

3 Union $\mathbf{a}_j^{pos}$ and $\mathbf{a}_j^{neg}$ to the entire selected vector ($\mathbf{a}_j$):
$\mathbf{a}_j = Concatenate(\mathbf{a}_j^{pos}, \mathbf{a}_j^{neg})$

4 Construct the specific rescaled LS_j^{tr} for the query input $\mathbf{x}_j^{tst}$:
$LS_j^{tr} = SelectLocalDataset(S^{tr}, \mathbf{a}_j)$

5 **Return** LS_j^{tr}.

Similarly, we can obtain num_{neg} nearest neighbors of the query input $\mathbf{x}_j^{tst}$ from S_{neg}^{tr} and the corresponding index vector, denoted as $\mathbf{a}_j^{neg} = [a_{j,1}^{neg}, \cdots, a_{j,i}^{neg}, \cdots, a_{j,|S_{neg}^{tr}|}^{neg}]^T$, by performing the following operation:

$$\mathbf{a}_j^{neg} = KNNSelection(\mathbf{x}_j^{tst}, S_{neg}^{tr}, num_{neg}) \tag{3}$$

We thus obtain K ($K = num_{pos} + num_{neg}$) nearest neighbors of the query input $\mathbf{x}_j^{tst}$ from S^{tr}, and the corresponding index vector $\mathbf{a}_j$ can be represented by concatenating $\mathbf{a}_j^{pos}$ and $\mathbf{a}_j^{neg}$ as follows:

$$\mathbf{a}_j = Concatenate(\mathbf{a}_j^{pos}, \mathbf{a}_j^{neg}) \tag{4}$$

Finally, the query-specific local training dataset LS_j^{tr} can be constructed from the original dataset S^{tr} according to the index vector $\mathbf{a}_j$:

$$LS_j^{tr} = SelectLocalDataset(S^{tr}, \mathbf{a}_j) \tag{5}$$

It should be noted that the underlying reason for replacing the parameter K in KNN with num_{pos} and num_{neg} is to flexibly control the ratio between the number of positive samples and the number of negative samples in the local training dataset LS_j^{tr}.

The entire DQD-RS algorithm is summarized in Algorithm 1.

Fig. 1 shows an example of the proposed DQD-SR procedure. Fig. 1(a) demonstrates a typical imbalanced sample distribution, where the white circles and black triangles represent examples of the majority and minority classes, respectively. In this example, we set the values of num_{pos} and num_{neg} to 5 and 6, respectively. Fig. 1(b) shows that the samples (circles and triangles in gray) of the majority and minority classes are selected by using the KNN algorithm according to the query input (the star), respectively. Fig. 1(c) shows the selected local training dataset for the query input.

2.2. Proposed ensemble scheme

The dataset generated with T-SR partially reflects the global distribution information of samples regardless of the characteristics of query input, whereas the dataset generated with DQD-SR partially reflects the local distribution information correlated to the query input. Thus, effectively ensembled classifiers trained with global and local information may potentially improve overall prediction performance. Previous studies [8,12,25,34,35] have also shown that ensembles of classifiers represent a promising route to improving prediction performance by utilizing complementary information. Thus, we present a new ensemble scheme that can ensemble the outputs of multiple classifiers.

Without loss of generality, we consider the binary classification problem. Suppose we have M trained classifiers. For a given query sample $\mathbf{x}_j^{tst}$, the mth ($1 \le m \le M$) classifier predicts a scalar output $p_j^m \in [0,1]$, which represents the confidence that $\mathbf{x}_j^{tst}$ belongs to the positive class. The ensembled output of the M classifiers under the query $\mathbf{x}_j^{tst}$ can therefore be expressed as follows:

$$p_j^{En} = \underset{p \in \{p_j^m\}_{m=1}^{M}}{\arg\max} |p - p_E| \tag{6}$$

where $p_E \in [0,1]$ is a prescribed ensemble parameter.

In essence, the proposed ensemble method is a much more general form than the traditional minimal ensemble and maximal ensemble, which are defined as follows:

$$p_j^{En} = \min_{p \in \{p_j^m\}_{m=1}^{M}} p \tag{7}$$

$$p_j^{En} = \max_{p \in \{p_j^m\}_{m=1}^{M}} p \tag{8}$$

For example, the proposed ensemble scheme (Eq. (6)) is equivalent to the traditional minimal ensemble (Eq. (7)) when $p_E = 1$, whereas the traditional maximal ensemble (Eq. (8)) is the

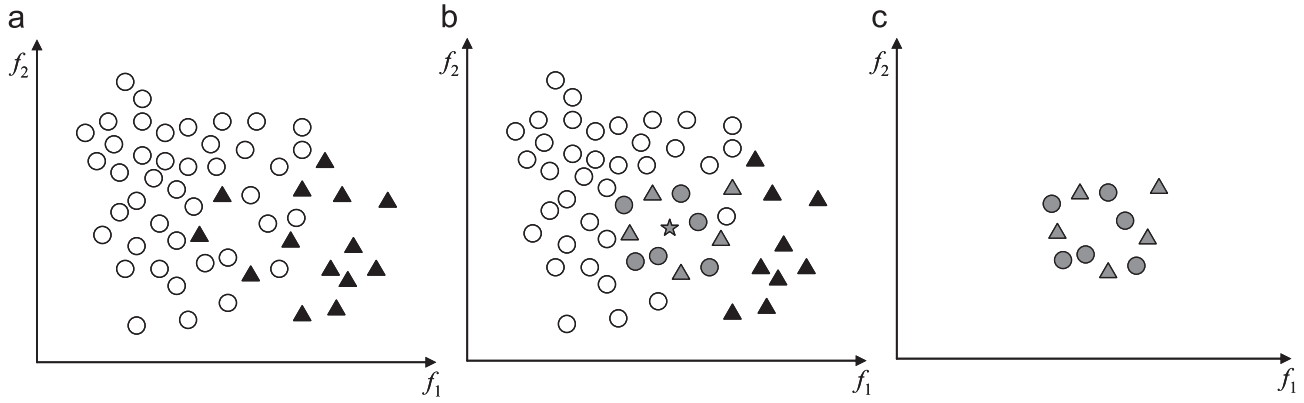


Fig. 1. (a) Original imbalanced training dataset; (b) selected samples according to the query input; and (c) selected local training dataset.

special case of the proposed ensemble scheme (Eq. (6)) when $p_E = 0$.

The manner in which the optimal ensemble parameter p_E is identified is further discussed in Section 3.7.

3. Materials and methods

3.1. Benchmark datasets

Three benchmark datasets collected by Chen et al. [33] were adopted to evaluate the efficacy of the proposed method.

The first dataset, NUC5tr, is a multiple nucleotide-binding dataset that consists of five training subsets, each for a special type of nucleotide. Specifically, NUC5tr consists of 227, 321, 140, 56, and 105 protein sequences (released in PDB before 10 March 2010) interacting with five types of nucleotides, i.e., ATP, ADP, AMP, GTP, and GDP, respectively. The maximal pairwise sequence identity of each of the five subsets was reduced to 40% with CD-hit [36].

The second dataset, NUC5tst, includes 17, 26, 20, 7, and 7 protein sequences (released in PDB after 10 March 2010) binding to ATP, ADP, AMP, GTP, and GDP, respectively. In this study, NUC5tst, which is independent of the NUC5tr, was prepared to demonstrate the generalizability of predicted models developed using NUC5tr. The maximal pairwise identity of the sequences of each type of nucleotide in NUC5tst is less than 40%; moreover, a sequence is removed from NUC5tst if it shares $>40\%$ identity to a protein in NUC5tr and both protein sequences bind to the same type of nucleotide.

The third dataset, denoted as nonNUC, contains 1372 protein sequences that do not interact with nucleotides. The nonNUC dataset is used to demonstrate whether the proposed method would "over-predict" protein–nucleotide binding residues. First, Chen et al. [33] used the PISCES server [37] to cull 1853 PDB chains, which correspond to high-quality 3D structures with a maximal R-factor of 0.25 and a maximal resolution of 1.6 Å at 20% sequence identity. The proteins that interact with nucleotides or share $>40\%$ identity to any chain in NUC5tr were then removed. Finally, the nonNUC, which contains 1372 sequences with 336,580 non-binding residues, was constructed. Table 1 summarizes the detailed compositions of the three benchmark datasets. All data listed in Table 1 can be found at http://csbio.njust.edu.cn/bioinf/TargetNUCs/Dataset.html. More details about the construction of the datasets can be found in [33].

Table 1
Compositions of the three benchmark datasets.

Dataset	Nucleotide type	No. of sequence	(n_pos, n_neg)[a]	Ratio[b]
NUC5tr	ATP	227	(3393, 80409)	24
	ADP	321	(4688, 121158)	26
	AMP	140	(1756, 44009)	25
	GTP	56	(875, 21401)	24
	GDP	105	(1577, 36561)	23
NUC5tst	ATP	17	(248, 6974)	28
	ADP	26	(405, 10553)	26
	AMP	20	(263, 6057)	23
	GTP	7	(134, 2678)	20
	GDP	7	(94, 2420)	26
nonNUC	–	1372	(0, 336580)	–

[a] Figures n_pos, n_neg in 2-tuple (n_pos, n_neg) represent the number of positive (binding residues) and negative (non-binding residues) samples, respectively.

[b] Ratio = n_neg/n_pos; and the symbol '–' means that the corresponding value does not exist.

3.2. Feature representation

Previous studies have demonstrated that many feature sources, such as position specific scoring matrix (PSSM), protein secondary structure (PSS), and amino acid physiochemical characteristics, can be effectively used for protein–nucleotide binding residue prediction [8,15,24,25,31–33,38]. Here, only PSSM and PSS are combined for the feature representation of residues because the purpose of this study is to investigate the efficacy of the proposed sample re-scaling strategy rather than to find the optimal feature representation for protein–nucleotide binding residue prediction.

3.2.1. Position Specific Scoring Matrix (PSSM)

For a given protein sequence with L amino acid residues, we generate its PSSM profile (L rows and 20 columns) by PSI-BLAST [39] searching against the Swiss-Prot database (composed of hundreds of millions of protein sequences) through three iterations with 10^{-3} as the E-value cutoff for multiple sequence alignment. Based on the original PSSM scores, we utilize the following logistic function (Eq. (9)) to further normalize each element in PSSM:

$$f(x) = \frac{1}{1+e^{-x}} \tag{9}$$

where x is the original element in PSSM. Then, motivated by [8,15,24,25,31–33,38], the sliding window technique is employed to extract the PSSM-view feature representation of each residue, i.e., the predicted nucleotide-binding probability is not only related to its PSSM scores but also to the PSSM scores of its neighboring residues with a window centered at the residue. In this study, we evaluated different sliding window sizes (ranging from 1 to 25 with a step size of 1) and found that 17 is the best choice. Accordingly, the dimensionality of the PSSM-view feature representation of a residue was $17 \times 20 = 340 - D$.

3.2.2. Protein secondary structure (PSS)

For a protein sequence with L residues, we obtained its PSS (L rows and 3 columns), which consists of three secondary structure classes (i.e., coil (C), helix (H), and strand (E)) for each residue, using the PSIPRED [40] software program. Again, a sliding window size of 17 was used to extract the PSS-view feature representation of each residue, and the dimensionality of the obtained feature vector of a residue was $17 \times 3 = 51 - D$.

The final feature representation of each residue was a $340+51=391-D$ feature vector, which was obtained by serially combining the residue's PSSM-view feature and the corresponding PSS-view feature.

3.3. Support vector machine

Support vector machine (SVM), a machine learning approach based on structural risk minimization principle of statistics learning theory [41], has been widely utilized in a wide variety of bioinformatics fields, including protein–nucleotide binding residue prediction [33]. Thus, we also employed SVM as the base model engine to evaluate the efficacy of our proposed imbalanced learning approach. In this study, we implemented SVM using LIBSVM software (version libsvm-3.18) [42], which can be downloaded for free at http://www.csie.ntu.edu.tw/~cjlin/libsvm/. The RBF kernel was selected to construct the kernel space. The regularization parameter (c) and the RBF kernel width parameter (γ) were optimized based on ten-fold cross-validation using a grid search strategy in the LIBSVM software [42] on the corresponding training dataset.

3.4. T-SR-based classifier

As described in Section 1, the T-SR (e.g., random under-sampling and random over-sampling) can be used as a basic strategy to reflect the global distribution information of an imbalanced dataset. Compared with random over-sampling (ROS), random under-sampling (RUS) can provide a much smaller and compact training dataset. Therefore, in this study, we employed the RUS algorithm as an example of the T-SR strategy to obtain the global distribution information of samples in the imbalanced protein-nucleotide interaction dataset.

RUS randomly removes a set of majority samples, denoted as R, from the original dataset $S = S_{min} \cup S_{maj}$ to relieve the degree of imbalance, where S_{min} and S_{maj} represent the minority and majority subsets, respectively, of the original dataset. After performing RUS, the obtained dataset can be represented as $\hat{S} = S_{min} \cup \hat{S}_{maj}$, where $\hat{S}_{maj} = S_{maj} - R$. We define $\alpha = |\hat{S}_{maj}| / |S_{min}|$ as the *imbalance coefficient* of a dataset, where $|\cdot|$ is the cardinality of the set. Clearly, the *imbalance coefficient* α reflects the severity of imbalance of a dataset, i.e., $\hat{S}$. If the value of α is equal to 1, we obtain a completely balanced under-sampled dataset from the original imbalanced dataset.

However, previous studies [8,38] have demonstrated that a completely balanced dataset (i.e., its $\alpha = 1$) does not definitely lead to the best performance. Better performance can be achieved by optimizing the value of the *imbalance coefficient* α, which is dataset-dependent parameter. Therefore, in this study, the optimal value of α was optimized by varying its value from 1 to 10 with a step size of 1 for each of the nucleotide-binding datasets.

Based on the optimized *imbalance coefficient* α^*, we could obtain a randomly under-sampled dataset $\hat{S} = S_{min} \cup \hat{S}_{maj}$. Then, we could train a base SVM classifier, denoted as *RUSModel*, on $\hat{S}$:

$$RUSModel = SVMTrain\left(\hat{S}\right) \tag{10}$$

For each query input (residue in study) $\mathbf{x}_j^{tst}$, the trained *RUSModel* could predict its probability of being nucleotide-binding, denoted as p_j^{RUS}, as follows:

$$p_j^{RUS} = SVMPredict\left(\mathbf{x}_j^{tst}, RUSModel_j\right) \tag{11}$$

3.5. DQD-SR-based classifier

For each query input (residue in study) $\mathbf{x}_j^{tst}$, we also dynamically trained a DQD-SR-based SVM classifier, denoted as $DQD-SRModel_j$, with the algorithm described in Section 2.1 as follows:

$$DQD-SRModel_j = SVMTrain(LS_j^{tr}) \tag{12}$$

The trained $DQD-SRModel_j$ embraces the local distribution knowledge correlated to the query input and can predict the probability of being nucleotide-binding, denoted as p_j^{DQD-SR}, for the query input.

$$p_j^{DQD-SR} = SVMPredict(\mathbf{x}_j^{tst}, DQD-SRModel_j) \tag{13}$$

3.6. Ensemble T-SR- and DQD-SR-based classifiers

To improve the over-prediction performance, the developed ensemble scheme (details in Section 2.2) was employed to ensemble classifiers trained with global and local information. Global and local information could be extracted from the dataset generated with T-SR and the dataset generated with DQD-SR, respectively. As described in Sections 3.4 and 3.5, p_j^{RUS} and p_j^{DQD-SR} denote the confidence outputs of the T-SR-based classifier and DQD-SR-based classifier for the jth testing sample $\mathbf{x}_j^{tst}$, respectively. Eq. (14) represents the developed ensemble scheme.

$$p_j^{En} = \underset{p \in \{p_j^{RUS}, p_j^{DQD-SR}\}}{\arg\max} |p - p_E| \tag{14}$$

where p_j^{En} is the final confidence of $\mathbf{x}_j^{tst}$ being classified into a positive class (i.e., nucleotide-binding residue), and $p_E \in [0, 1]$ is the prescribed ensemble parameter. How the optimal ensemble parameter p_E is identified is further discussed in the following section.

3.7. Evaluation Indices

In this study, five evaluation indices routinely used in this field (i.e., Specificity (*Spe*), Sensitivity (*Sen*), Precision (*Pre*), Accuracy (*Acc*), and the Mathew's Correlation Coefficient (*MCC*)) were utilized to demonstrate the prediction quality of the proposed method as follows:

$$Spe = \frac{TN}{TN+FP} \tag{15}$$

$$Sen = \frac{TP}{TP+FN} \tag{16}$$

$$Pre = \frac{TP}{TP+FP} \tag{17}$$

$$Acc = \frac{TP+TN}{TP+FN+TN+FP} \tag{18}$$

$$MCC = \frac{TP \cdot TN - FN \cdot FP}{\sqrt{(TP+FN) \cdot (TP+FP) \cdot (TN+FN) \cdot (TN+FP)}} \tag{19}$$

where *TN*, *TP*, *FN*, and *FP* represent true negative, true positive, false negative, and false positive, respectively.

For a soft-type classifier (e.g., T-SR-based classifier, DQD-SR-based classifier, and the ensemble classifier used in this study) that outputs a continuous numerical value to represent the confidence of a sample belonging to the predicted class, gradually adjusting the classification threshold will produce a series of confusion matrices [43]. From each confusion matrix, the corresponding indices *Spe*, *Sen*, *Pre*, *Acc*, and *MCC* can then be computed. In other words, these five evaluation indices are threshold dependent. Therefore, determining how to objectively report these evaluation indices is a significant problem, especially in an imbalanced learning scenario. In general, people would expect that a predictor can provide high accuracy for the minority class (e.g., binding residues in this study) without severely jeopardizing the accuracy of the majority class (non-binding residues) [6].

In view of the above-mentioned issues, together with the fact that *MCC* provides the overall measurement of the quality of binary predictions, we chose the threshold, denoted as T, that maximizes the value of *MCC* of the predictions to report the above-mentioned five indices. This strategy has been adopted by many other protein attribute predictors, including protein–nucleotide binding residue predictors. More specifically, we first identified the threshold that maximizes the value of *MCC* of the predictions on each training subset of NUC5tr using cross-validation, and then the identified threshold (rather than another optimized one) was used to evaluate the performance of the proposed method on the corresponding independent validation subset of NUC5tst as well as nonNUC dataset.

Another issue that should be addressed is determining how to identify the optimal ensemble parameter p_E for the ensemble classifier (Eq. (14)). We gradually varied the value of p_E from 0 to 1 with a step size of 0.01; for each value of p_E, we calculated the maximal value of *MCC* of the ensembled classifier that could be

obtained on the training dataset over cross-validation; the optimal p_E was the one that maximized the value of *MCC*.

4. Experimental results and analysis

4.1. Choosing the value of the Imbalance Coefficient to train an RUS-based classifier

In this section, we will try to empirically choose an appropriate value for the imbalance coefficient (α) to train an RUS-based classifier. For each of the five types of nucleotide, i.e., ATP, ADP, AMP, GTP, and GDP, we evaluated the *MCC* performance variations of an RUS-based SVM classifier on the corresponding training subset over five-fold cross-validation by gradually varying the value of α from 1 to 10 with a step size of 1.

Fig. 2 plots the performance variation curves of *MCC* versus α for the five nucleotides, whereas Table 2 summarizes the detailed *MCC* values for each type of nucleotide with different values of α.

Fig. 2 and Table 2 clearly show that the overall trend exhibited by the values of *MCC* tends to increase with α when $\alpha \leq 5$, indicating a distinct enhancement for all five types of nucleotide. When $5 < \alpha \leq 8$, the value of *MCC* for each type of nucleotide slightly increases. However, when $\alpha > 8$, the values of *MCC* tend to decrease for almost all of the nucleotides. It can be expected that the performance of *MCC* will further deteriorate with the increase in α when $\alpha > 10$ because the severity of imbalance will become increasingly serious with the increase in α. Thus, in all subsequent experiments, we set $\alpha = 8$ to train an RUS-based classifier.

4.2. Setting the values of num_{pos} and num_{neg} for training a DQD-SR-based classifier

To obtain a relatively compact local training dataset for a given query input when applying DQD-SR, we suggest that users choose two relatively small values for parameters num_{pos} and num_{neg}. We optimized the values of num_{pos} and num_{neg} with a grid search strategy. More specifically, we varied the value of num_{pos} from 10 to 100 with a step size of 10 and varied the value of num_{neg} from 50 to 500 with a step size of 50. For each type of nucleotide, we calculated the *MCC* variations of the DQD-SR-based SVM classifier on the corresponding training subset over five-fold cross-validation with different (num_{pos}, num_{neg}) pairs. According to the experimental results, $num_{pos} = 50$ and $num_{neg} = 300$ are the best choices for the five nucleotides.

We also analyzed the robustness of the proposed DQD-SR on the two key parameters, i.e., num_{pos} and num_{neg}, using the following two methods.

First, we fixed $num_{pos} = 50$ (the optimal value we obtained by grid search) and varied the value of num_{neg} from 50 to 500 with a step size of 50. Then, for each of the five nucleotides, we evaluated the *MCC* performance variations of a DQD-SR-based SVM classifier on the corresponding training subset over five-fold cross-validation with different values of num_{neg}.

Fig. 3 plots the performance variation curves of *MCC* versus num_{neg} for the five nucleotide training datasets over five-fold cross-validation, whereas Table 3 summarizes the detailed *MCC* values for each type of nucleotide with different values of num_{neg}.

Fig. 3 and Table 3 show that the *MCC* values tend to increase with the increase in num_{neg} when $50 \leq num_{neg} \leq 250$; the *MCC* values reach a peak when $num_{neg} = 300$; when $num_{neg} > 300$, the *MCC* values slightly fluctuate and enter a slowly descending state. In other words, the proposed DQD-SR-based SVM classifier is relatively robust to the parameter num_{neg} on all five nucleotide datasets when $num_{neg} \geq 300$. Similarly, we could analyze the robustness of the proposed method to the parameter num_{pos}. Due to space limitations, we do not provide the detailed results.

Second, we further analyzed the robustness of the proposed method by gradually increasing the value of $num_{pos} + num_{neg}$ from 20 to 200 with a step size of 20 with $num_{pos} = num_{neg}$. As an example, we plotted the performance variation curves of *MCC* versus $num_{pos} + num_{neg}$ of the proposed method on the GTP

Table 2
The detailed *MCC* values for each type of nucleotide with different values of α.

α	ATP	ADP	AMP	GTP	GDP
1	0.457	0.564	0.352	0.521	0.697
2	0.487	0.591	0.387	0.534	0.725
3	0.506	0.604	0.421	0.553	0.739
4	0.522	0.612	0.423	0.559	0.750
5	0.532	0.618	0.433	0.560	0.753
6	0.536	0.624	0.438	0.559	0.755
7	0.543	0.627	0.441	0.566	0.757
8	**0.551**	**0.637**	**0.455**	0.568	**0.761**
9	0.547	0.633	0.454	**0.571**	0.760
10	0.546	0.630	0.449	0.565	0.759

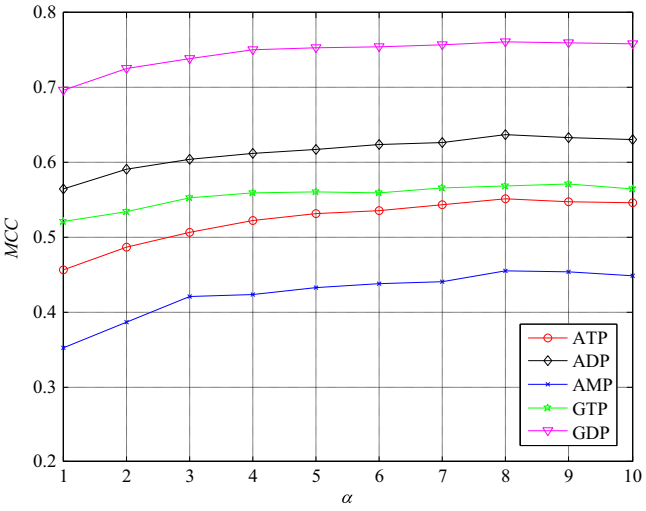


Fig. 2. The performance variation curves of *MCC* versus α for the five nucleotides ATP, ADP, AMP, GTP, and GDP.

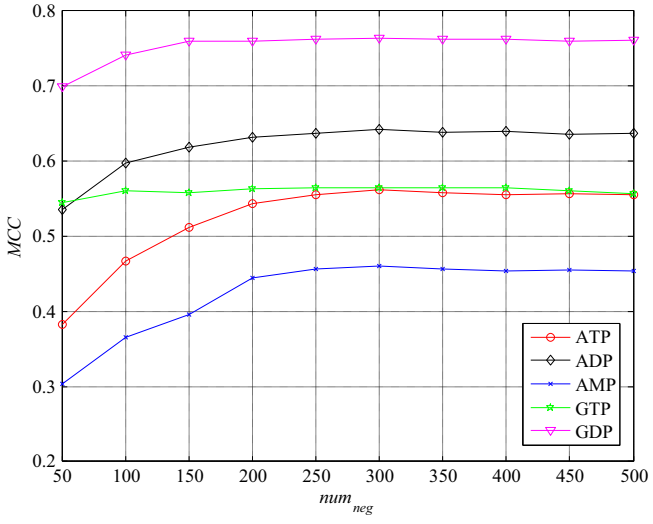


Fig. 3. The performance variation curves of *MCC* versus num_{neg} with fixed $num_{pos} = 50$ for the five nucleotides ATP, ADP, AMP, GTP, and GDP.

Table 3
The detailed *MCC* values for each type of nucleotide with different values of num_{neg} with $num_{pos} = 50$.

num_{neg}	ATP	ADP	AMP	GTP	GDP
50	0.383	0.535	0.303	0.545	0.699
100	0.467	0.598	0.366	0.561	0.741
150	0.512	0.619	0.396	0.558	0.760
200	0.543	0.632	0.444	0.563	0.760
250	0.556	0.637	0.457	**0.564**	0.763
300	**0.562**	**0.643**	**0.460**	**0.564**	**0.764**
350	0.558	0.639	0.456	**0.564**	0.762
400	0.556	0.640	0.454	**0.564**	0.763
450	0.557	0.636	0.455	0.560	0.760
500	0.555	0.637	0.454	0.557	0.761

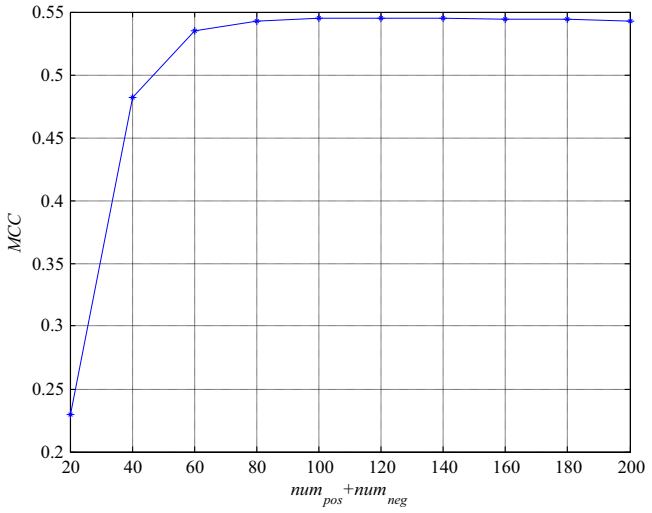


Fig. 4. The performance variation curve of *MCC* versus $num_{pos}+num_{neg}$ under $num_{pos} = num_{neg}$ on the GDP dataset.

dataset over five-fold cross-validation (Fig. 4). The figure clearly shows that the proposed method is robust with respect to *MCC* when $num_{pos}+num_{neg}$ is sufficiently large (80 in this experiment).

4.3. Improving performance by ensembling T-SR- and DQD-SR-based classifiers

To verify that the developed ensemble scheme is useful, we compared the performances of the T-SR-based classifier, DQD-SR-based classifier, and ensembled classifier. Fig. 5 shows the details of the performance comparisons.

Fig. 5 shows that the ensembled classifier almost always outperforms the T-SR- and DQD-SR-based classifiers with respect to *Sen*, *ACC*, *Pre*, and *MCC* for all five nucleotide subsets. For example, the values of *MCC* for the ensembled classifier for ATP, ADP, AMP, GTP, and GDP on the corresponding datasets are 0.573, 0.656, 0.474, 0.591, and 0.771, which are 2.2%, 1.9%, 1.9%, 2.3%, and 1.0% higher than the values measured for the T-SR-based classifier and 1.1%, 1.3%, 1.4%, 2.7%, and 0.7% higher than those measured for the DQD-SR-based classifier.

Fig. 5 suggests that the proposed ensemble scheme does help improve the prediction performance of the individual base classifiers (e.g., the T-SR- and DQD-SR-based classifiers in this study).

4.4. Comparisons with other protein–nucleotide binding residue predictors

In this section, we compare the proposed method, called TargetNUCs, with other popular sequence-based protein–nucleotide binding residue predictors, including SVMPred [32], Rate4site [44], NsitePred [33], TargetATP [8], TargetATPsite [24], and TargetS [25], to demonstrate its efficacy. It should be noted that that TargetNUCs ensembles an RUS-based SVM classifier and a DQD-SR-based SVM classifier, both with the combined PSSM and PSS features as model input. To make strict comparisons, we performed cross-validation tests, independent validation tests, and non-nucleotide

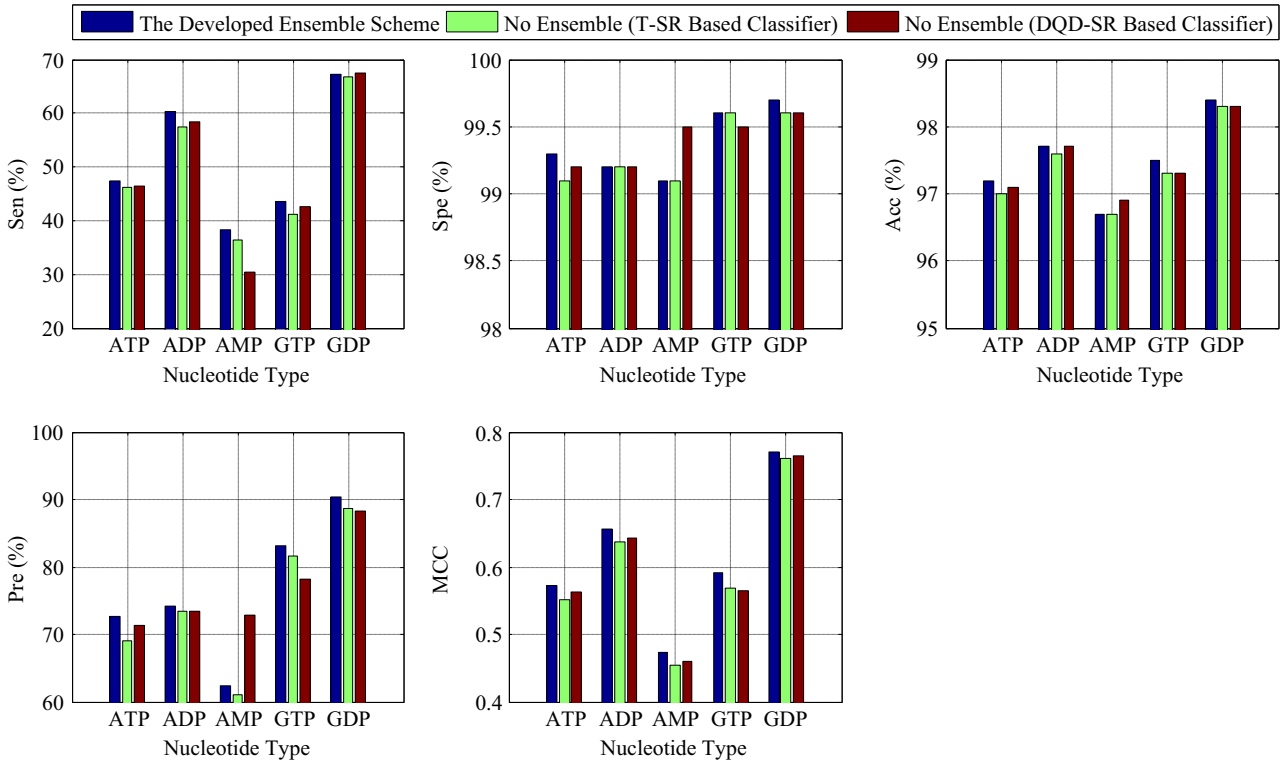


Fig. 5. Performance comparison of the developed ensemble scheme, T-SR-based classifier, and DQD-SR-based classifier on NUC5tr over five-fold cross-validation.

Table 4
Performance comparisons of the proposed TargetNUCs with other protein–nucleotide binding residues predictors on NUC5tr over five-fold cross-validation tests.

Type	Predictor	Sen (%)	Spe (%)	Acc (%)	Pre (%)	MCC
ATP	TargetNUCs ($T=0.23$)	**47.3**	**99.3**	**97.2**	**72.7**	**0.573**
	TargetS [25]	44.6	99.0	96.7	64.0	0.531
	TargetATPsite [24]	44.5	98.9	96.6	62.1	0.520
	TargetATP [8]	41.2	99.0	96.6	62.8	0.501
	NsitePred [a]	44.4	98.2	96.0	51.9	0.460
	SVMPred [a]	36.1	98.8	96.2	56.4	0.433
	Rate4site [a]	44.6	87.0	85.2	13.2	0.182
ADP	TargetNUCs ($T=0.12$)	**60.2**	99.2	**97.7**	**74.1**	**0.656**
	TargetS [25]	58.7	99.0	97.5	69.4	0.631
	NsitePred [a]	54.4	98.8	97.1	63.3	0.572
	SVMPred [a]	45.8	**99.3**	97.3	70.4	0.555
	Rate4site [a]	47.2	84.4	83.0	10.6	0.161
AMP	TargetNUCs ($T=0.14$)	38.3	99.1	**96.7**	62.4	**0.474**
	TargetS [25]	36.8	98.6	96.1	50.1	0.418
	NsitePred [a]	30.4	98.8	96.2	51.1	0.377
	SVMPred [a]	20.8	**99.6**	96.6	**66.7**	0.360
	Rate4site [a]	**56.2**	79.9	79.0	10.7	0.174
GTP	TargetNUCs ($T=0.22$)	43.6	99.6	**97.5**	83.2	0.591
	TargetS [25]	44.3	99.6	97.4	81.7	**0.595**
	NsitePred [a]	47.3	99.1	96.8	70.6	0.562
	SVMPred [a]	37.3	**99.7**	97.0	**84.8**	0.551
	Rate4site [a]	**56.9**	80.6	79.6	10.8	0.180
GDP	TargetNUCs ($T=0.14$)	**67.2**	**99.7**	**98.4**	**90.4**	**0.771**
	TargetS [25]	65.0	99.6	98.1	87.5	0.741
	NsitePred [a]	64.6	99.1	97.6	73.4	0.675
	SVMPred [a]	62.3	98.9	97.7	71.6	0.655
	Rate4site [a]	51.6	82.3	81.1	11.0	0.170

[a] Data excerpted from [33].

binding tests on the NUC5tr, NUC5tst, and nonNUC datasets, respectively.

4.4.1. Performance comparisons over the cross-validation tests

Table 4 lists the performance comparisons between the proposed TargetNUCs predictor and the TargetS [25], TargetATPsite [24], TargetATP [8], NsitePred [33], SVMPred [32], and Rate4site [44] predictors with respect to the five training subsets of NUC5tr over five-fold cross-validation. Table 4 shows that the TargetNUCs yields $Pre > 0.62$ and $MCC > 0.47$ for all five nucleotide types. The *MCC* values of TargetNUCs are consistently superior to those of the other predictors, i.e., TargetS [25], TargetATPsite [24], TargetATP [8], NsitePred [33], SVMPred and Rate4site [44], for four out of the five considered nucleotide types (except GTP), and an average enhancement of 3.8% is achieved relative to the second best performer, TargetS [25]. Note that TargetS [25], TargetATPsite [24], and TargetATP [8] listed in Table 4 are the three most recently released predictors developed by our group. We also observed that TargetNUCs significantly outperforms three other predictors, i.e., NsitePred [33], SVMPred [32], and Rate4site [44]. Taking NsitePred [33] as an example, TargetNUCs achieves average improvements of 0.58%, 0.76%, 14.5%, and 8.38% with respect to *Spe*, *Acc*, *Pre*, and *MCC*, respectively, for the five nucleotides. Furthermore, TargetNUCs serves as the second best predictor for GTP with respect to the *MCC* index, whose value is slightly lower than that yielded by our TargetS predictor [25]. We speculate that the main reason for this finding may be that the sample distribution information of GTP is not sufficient because there are only 56 sequences in the GTP training subset.

We acknowledge that SVMPred [32] achieves the best performance with respect to *Spe* for several nucleotide types (i.e., 0.993, 0.996, and 0.997 for ADP, AMP, and GTP, respectively) and the highest values for *Pre* for AMP and GTP, i.e., 66.7% and 84.8%. However, the *Sen* values obtained by SVMPred are the lowest, implying that too many false negatives are produced during prediction. On the other hand, Rate4site [44] achieves the highest *Sen* values (56.9% and 56.2% for GTP and AMP, respectively). However, the *Spe* values of Rate4site [44] are much lower, i.e., too many false positives are incurred during prediction.

Table 5
Performance comparison of the proposed TargetNUCs with other protein–nucleotide binding residues predictors over independent validation tests.

Type	Predictor	Sen (%)	Spe (%)	Acc (%)	Pre (%)	MCC
ATP	TargetNUCs ($T=0.23$)	**51.6**	**99.2**	**97.5**	**68.8**	**0.584**
	TargetS [a]	50.4	98.9	97.2	60.1	0.534
	TargetATPsite [24]	45.8	99.1	97.2	63.6	0.530
	TargetATP [8]	48.9	98.9	96.9	57.2	0.542
	NsitePred [b]	46.0	98.5	96.7	52.8	0.476
	SVMPred [b]	36.7	99.1	96.9	58.7	0.451
	Rate4site [b]	46.4	86.2	84.9	10.7	0.167
ADP	TargetNUCs ($T=0.12$)	**55.2**	**98.7**	**97.1**	61.4	**0.567**
	TargetS [a]	50.9	98.5	96.8	55.7	0.516
	NsitePred [b]	47.4	**98.7**	96.8	58.9	0.512
	SVMPred [b]	38.8	99.3	**97.1**	**68.0**	0.500
	Rate4site [b]	52.1	82.3	81.2	10.2	0.166
AMP	TargetNUCs ($T=0.14$)	43.0	99.0	96.7	66.1	**0.517**
	TargetS [a]	44.5	98.7	96.5	60.9	0.503
	NsitePred [b]	42.3	98.7	**96.9**	60.6	0.501
	SVMPred [b]	33.5	**99.4**	96.7	**72.1**	0.478
	Rate4site [b]	**52.0**	82.4	81.1	11.4	0.175
GTP	TargetNUCs ($T=0.22$)	57.5	**99.7**	**97.7**	**89.5**	**0.707**
	TargetS [a]	**62.6**	98.7	97.0	71.2	0.653
	NsitePred [b]	60.4	98.8	96.9	71.1	0.640
	SVMPred [b]	48.5	99.3	96.9	78.3	0.602
	Rate4site [b]	53.1	81.7	80.6	10.3	0.168
GDP	TargetNUCs ($T=0.14$)	46.8	**99.5**	**97.5**	**77.2**	**0.590**
	TargetS [a]	45.9	99.4	97.4	74.2	0.571
	NsitePred [b]	**58.5**	98.4	97.0	59.8	0.576
	SVMPred [b]	51.1	98.8	97.1	63.2	0.553
	Rate4site [b]	54.5	79.3	78.1	11.6	0.173

[a] Results computed using the TargetS [25] models which are trained on the corresponding dataset of NUC5tr.
[b] Data excerpted from [33].

4.4.2. Performance comparisons over independent validation tests

Independent validation tests are often applied as a mandatory procedure to appraise the generalization capability of a trained predictor. For this reason, we also performed independent validation tests. First, we trained TargetNUCs on each of the five training subsets of NUC5tr; then, the trained TargetNUCs was tested with the corresponding independent validation subsets listed in Table 1. Note that to fairly compare TargetNUCs with other predictors and avoid potential over-estimation of TargetNUCs, the same threshold identified during the five-fold cross-validation test on each of the five training subsets was applied to perform the corresponding independent validation test.

Table 5 summarizes the performance comparisons between the TargetNUCs and other popular protein–nucleotide binding residue predictors on the five independent validation subsets of NUC5tst. The table shows that TargetNUCs outperformed the other considered predictors and served as the best performer with respect to the *MCC* evaluation index. The *MCC* values of ATP, ADP, AMP, GTP, and GDP were 0.584, 0.567, 0.517, 0.707, and 0.590, making them approximately 4.2%, 5.1%, 1.4%, 5.4%, and 1.4% better than the *MCC* values yielded by the second best predictors, respectively. Careful analysis of Table 5 also reveals that the proposed TargetNUCs performs the best for three nucleotide types (i.e., ATP, GTP, and GDP) with respect to the *Pre* evaluation index. An average improvement of 6.46% on *Pre* is observed when compared with the second best predictor on ATP, GTP, and GDP. By revisiting Table 4, we find that the *MCC* values of ATP, ADP, AMP, GTP, and GDP on the corresponding training subsets of NUC5tr over five-fold cross-validation are 0.573, 0.656, 0.474, 0.591, and 0.771, respectively. In

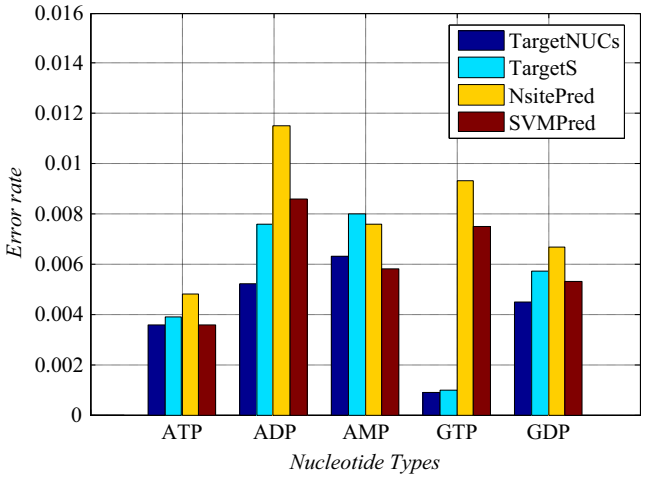


Fig. 6. The error rates of TargetNUCs and the other considered predictors on the non-nucleotide binding dataset nonNUC.

other words, for all five nucleotide types, TargetNUCs obtained similar *MCC* values on the training dataset (NUC5tr) and independent validation dataset (NUC5tst), meaning that the generalizability of the TargetNUCs derived from the knowledge buried in training datasets has not been over-fitted or under-estimated.

4.4.3. Performance comparisons over non-nucleotide binding tests

We also performed performance comparisons between the proposed TargetNUCs and the other methods, i.e., TargetS [25], NsitePred [33], and SVMPred [32], on protein sequences that do not interact with nucleotides (i.e., non-nucleotide binding sequences in nonNUC dataset). Fig. 6 plots the error rate, which is defined as the ratio between the number of false positives (i.e., the residues that are mistakenly predicted as binding residues) and the total number of residues for each of the considered predictors on nonNUC dataset. Fig. 6 shows that the proposed TargetNUCs achieved the best performance for four out of the five considered nucleotides: the error rates of TargetNUCs for ATP, ADP, GTP, and GDP are only 0.36%, 0.52%, 0.09%, and 0.45%, respectively, which are the lowest among the four considered predictors. Regarding AMP, TargetNUCs showed worse performance than SVMPred but still performed better than the other two predictors, i.e., TargetS and NsitePred.

5. Conclusions

In this study, a new query-driven sample rescaling strategy called DQD-SR, which dynamically obtains the local sample distribution information of a query input, was proposed for class imbalanced learning. We also proposed a decision-level ensemble scheme to effectively integrate the global sample distribution information and the local sample distribution information obtained from T-SR and DQD-SR, respectively. To demonstrate the effectiveness of the proposed method for imbalanced learning, we applied the method to protein–nucleotide binding residue prediction, which is a typical imbalanced learning problem in bioinformatics. Stringent cross-validation tests and independent validation tests on benchmark datasets demonstrate the efficacy of the proposed method. Furthermore, to reap the full benefits of the proposed method, we implemented an on-line webserver, called TargetNUCs, for protein–nucleotide residue prediction.

Because the current version of DQD-SR utilizes the traditional KNN algorithm to reveal the local sample distribution information of a query input, time consumption is still a challenging problem, especially in cases in which a large number of samples are involved. Nevertheless, the proposed method may help solve some problems associated with traditional static models [26]:

(1) *Low scalability*: Traditional static methods often suffer from the disadvantage of low scalability; thus, the trained static model has to be continuously re-trained to encompass the knowledge contained in the new annotated data. The proposed DQD-SR can quickly accommodate new annotated data and thus possesses better scalability: When new annotated data are available, the method simply adds them to the existing dataset, and no additional work is required. Whether the new data will be used for prediction depends on the query input.
(2) *Impractical when the dataset is large*: When a dataset is sufficiently large, training and optimizing a static model on the entire dataset is often impractical. Unlike static methods, which train prediction models on the entire dataset, the proposed DQD-SR dynamically trains a compact prediction model with a compact, query-driven dataset. This feature is especially useful when the dataset is large, in which case training a static prediction model with the entire dataset is not possible.

In our future work, we aim to improve the time efficiency of DQD-SR. Two possible solutions are as follows:

(1) Apply modified KNN algorithms to accelerate computation. When the dataset is enormous and the traditional KNN is not applicable, fast KNN algorithms such as TFKNN [45], NN-Descent [46], and SMART-TV [47] could be used to precompute the localization. Taking TFKNN [45] as an example, a B+ tree can be constructed in advance to facilitate the subsequent fast localization of the neighbor set of a query input.
(2) Replace KNN with other fast-searching algorithms, such as hash indexing [48,49], mountain-climb searching [50], and fast tag searching protocol [51].

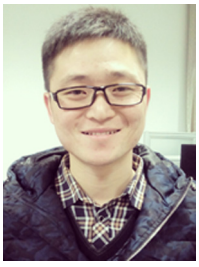
Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61373062 and 61222306) and the Natural Science Foundation of Jiangsu Province (No. BK20141403).

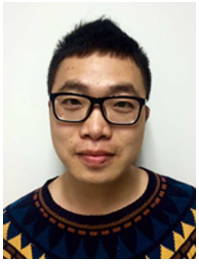
References

[1] Z. Lin, Z. Hao, X. Yang, X. Liu, Several SVM ensemble methods integrated with under-sampling for imbalanced data learning, Advanced Data Mining and Applications, Springer (2009), p. 536–544.
[2] I. Mani, I. Zhang, kNN approach to unbalanced data distributions: a case study involving information extraction, in: Proceedings of Workshop on Learning from Imbalanced Datasets, 2003.
[3] S. Wang, L.L. Minku, X. Yao, A learning framework for online class imbalance learning, Comput. Intell. Ensemble Learn. (2013) 36–45.
[4] P. Yang, P.D. Yoo, J. Fernando, B.B. Zhou, Z. Zhang, A.Y. Zomaya, Sample subset optimization techniques for imbalanced and ensemble learning problems in bioinformatics applications, IEEE Trans. Cybern. 44 (2013) 445–455.
[5] S. Barua, M.M. Islam, X. Yao, K. Murase, MWMOTE—majority weighted minority oversampling technique for imbalanced data set learning, IEEE Trans. Knowl. Data Eng. 26 (2014) 405–425.
[6] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (2009) 1263–1284.
[7] S. Maldonado, J. López, Imbalanced data classification using second-order cone programming support vector machines, Pattern Recognit. 47 (2014) 2070–2079.
[8] D.J. Yu, J. Hu, Z.M. Tang, H.B. Shen, J. Yang, J.Y. Yang, Improving protein–ATP binding residues prediction by boosting SVMs with random under-sampling, Neurocomputing 104 (2013) 180–190.

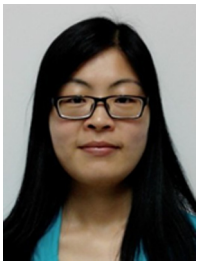
[9] H. Guo, Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach, SIGKDD Explor. 6 (2004) 2004.
[10] C.-Y. Yang, J.-S. Yang, J.-J. Wang, Margin calibration in SVM class-imbalanced learning, Neurocomputing 73 (2009) 397–411.
[11] G.M. Weiss, Mining with rarity: a unifying framework, ACM SIGKDD Explor. Newsl. 6 (2004) 7–19.
[12] L. Nanni, A novel ensemble of classifiers for protein fold recognition, Neurocomputing 69 (2006) 2434–2437.
[13] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[14] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: adaptive synthetic sampling approach for imbalanced learning, Neural Netw. (2008) 1322–1328.
[15] J. Hu, X. He, D.-J. Yu, X.-B. Yang, J.-Y. Yang, H.-B. Shen, A new supervised over-sampling algorithm with application to protein–nucleotide binding residue prediction, PLoS One 9 (2014) e107676.
[16] N. Japkowicz, C. Myers, M. Gluck, A Novelty Detection Approach to Classification, in: Proceedings of the International Joint Conference on Artificial Intelligence, 1995, pp. 518–523.
[17] Z.H. Zhou, X.Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Trans. Knowl. Data Eng. 18 (2006) 63–77.
[18] L. Rokach, Ensemble-based classifiers, Artif. Intell. Rev. 33 (2010) 1–39.
[19] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: improving prediction of the minority class in boosting, Lect. Notes Comput. Sci. (2003) 107–119.
[20] S. Wang, H. Chen, X. Yao, Negative correlation learning for classification ensembles, in: Proceedings of the 2010 International Joint Conference on NeuralNetworks (IJCNN), 2010, pp. 1–8.
[21] S. Wang, X. Yao, The effectiveness of a new negative correlation learning algorithm for classification ensembles, in: Proceedings of the 2010 IEEE International Conference on Data Mining Workshops (ICDMW), 2010, pp. 1013–1020.
[22] G.M. Weiss, Provost, learning when training data are costly: the effect of class distribution on tree induction, J. Artif. Intell. Res. 19 (2003) 315–354.
[23] R.C. Holte, L. Acker, B.W. Porter, Concept learning and the problem of small disjuncts, in: Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 1989, pp. 813–818.
[24] D.J. Yu, J. Hu, Y. Huang, H.B. Shen, Y. Qi, Z.M. Tang, J.Y. Yang, TargetATPsite: a template-free method for ATP-binding sites prediction with residue evolution image sparse representation and classifier ensemble, J. Comput. Chem. 34 (2013) 974–985.
[25] D.J. Yu, J. Hu, J. Yang, H.B. Shen, J.H. Tang, J.Y. Yang, Designing template-free predictor for targeting protein-ligand binding sites with classifier ensemble and spatial clustering, IEEE-ACM Trans, IEEE-ACM Trans. Comput. Biol. Bioinform. 10 (2013) 994–1008.
[26] D.-J. Yu, J. Hu, Q.-M. Li, Z.-M. Tang, J.-Y. Yang, H.-B. Shen, Constructing Query-driven dynamic machine learning model with application to protein-ligand binding sites prediction, IEEE Trans. NanoBiosci. 14 (2015) 45–58.
[27] D.T. Larose, *k*-Nearest Neighbor Algorithm, Discovering Knowledge in Data: An Introduction to Data Mining, 2005, pp. 90–106.
[28] B. Alberts, Molecular Biology of the Cell, 5th Ed, Garland Science, New York, 2008.
[29] M. Gao, J. Skolnick, The distribution of ligand-binding pockets around protein-protein interfaces suggests a general mechanism for pocket formation, Proc. Natl. Acad. Sci. USA 109 (2012) 3784–3789.
[30] H. Kokubo, T. Tanaka, Y. Okamoto, Ab initio prediction of protein-ligand binding structures by replica-exchange umbrella sampling simulations, J. Comput. Chem. 32 (2011) 2810–2821.
[31] J.S. Chauhan, N.K. Mishra, G.P. Raghava, Identification of ATP binding residues of a protein from its primary sequence, BMC Bioinform. 10 (2009) 434.
[32] K. Chen, M.J. Mizianty, L. Kurgan, ATPsite: sequence-based prediction of ATP-binding residues, Proteome Sci. 9 (Suppl. 1) (2011) S4.
[33] K. Chen, M.J. Mizianty, L. Kurgan, Prediction and analysis of nucleotide-binding residues using sequence and sequence-derived structural descriptors, Bioinformatics 28 (2012) 331–341.
[34] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 39 (2009) 539–550.
[35] Q. Gu, Y.-S. Ding, T.-L. Zhang, An ensemble classifier based prediction of G-protein-coupled receptor classes in low homology, Neurocomputing (2015).
[36] W. Li, A. Godzik, Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences, Bioinformatics 22 (2006) 1658–1659.
[37] G. Wang, R.L. Dunbrack Jr., PISCES: a protein sequence culling server, Bioinformatics 19 (2003) 1589–1591.
[38] Y.N. Zhang, D.J. Yu, S.S. Li, Y.X. Fan, Y. Huang, H.B. Shen, Predicting protein-ATP binding sites from primary sequence through fusing bi-profile sampling of multi-view features, BMC Bioinform. 13 (2012) 118.
[39] A.A. Schaffer, L. Aravind, T.L. Madden, S. Shavirin, J.L. Spouge, Y.I. Wolf, E. V. Koonin, S.F. Altschul, Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements, Nucleic Acids Res. 29 (2001) 2994–3005.
[40] D.T. Jones, Protein secondary structure prediction based on position-specific scoring matrices, J. Mol. Biol. 292 (1999) 195–202.
[41] V. Vapnik, The Nature of Statistical Learning Theory, Springer Science & Business Media, 2000.
[42] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology ((TIST)) 2 (2011) 27.
[43] H. Haibo, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (2009) 1263–1284.
[44] T. Pupko, R.E. Bell, I. Mayrose, F. Glaser, N. Ben-Tal, Rate4Site: an algorithmic tool for the identification of functional regions in proteins by surface mapping of evolutionary determinants within their homologues, Bioinformatics 18 (Suppl. 1) (2002) S71–S77.
[45] Y. Wang, Z.-O. Wang, A fast KNN algorithm for text categorization, Mach. Learn. Cybern. (2007) 3436–3441.
[46] W. Dong, C. Moses, K. Li, Efficient *k*-nearest neighbor graph construction for generic similarity measures, in: Proceedings of the 20th international conference on World wide web, (ACM2011), pp. 577–586.
[47] T. Abidin, W. Perrizo, SMART-TV: A fast and scalable nearest neighbor based classifier for data mining, in: Proceedings of the 2006 ACM symposium on Applied computing (ACM2006), pp. 536–540.
[48] K. Grauman, R. Fergus, Learning Binary has Codes for Large-scale Image Search, Machine for Computer vision, Springer, 2013, pp. 49–87.
[49] F. Yue, B. Li, M. Yu, J. Wang, Hashing based fast palmprint identification for large-scale databases, IEEE Trans. Inf. Forensics Secur. 8 (2013) 769–778.
[50] P. Wencheng, Theory and application of parameter self-optimizing intelligent sampling method, in: 2010 IEEE 10th International Conference on Signal Processing (ICSP) (IEEE2010), 2010, pp. 66–69.
[51] Y. Zheng, M. Li, Fast tag searching protocol for large-scale RFID systems, IEEE/ACM Trans. Netw. 21 (2013) 924–934.



Jun Hu received his B.S. degree in computer science from the Anhui Normal University, China in 2011. Currently, he is working towards the Ph.D. degree in the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His research interests include bioinformatics, data mining, and pattern recognition.



Yang Li received the B.S. degree in computer science from the Nanjing University of Science and Technology, China in 2014. He is currently working toward the Ph.D. degree in the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His research interests include bioinformatics, data mining, and pattern recognition.



Wu-Xia Yan received her B.S. and M.S. degrees in 2008 and 2011, respectively, and she is currently pursuing her Ph.D. degree in the School of Computer Science and Engineering, Nanjing University of Science and Technology (NUST). Her current research interests include image processing and computer vision.

Jing-Yu Yang received the B.S. Degree in Computer Science from NUST, Nanjing, China. From 1982 to 1984 he was a visiting scientist at the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. From 1993 to 1994 he was a visiting professor at the Department of Computer Science, Missouri University. And in 1998, he acted as a visiting professor at the Concordia University in Canada. He is currently a professor and Chairman in the Department of Computer Science at NUST. He is the author of over 150 scientific papers in computer vision, pattern recognition, and artificial intelligence. He has won more than 20 provincial awards and national awards. His current research interests are in the areas of pattern recognition, robot vision, image processing, data fusion, and artificial intelligence.

Hong-Bin Shen received his Ph.D. degree from Shanghai Jiaotong University China in 2007. He was a postdoctoral research fellow of Harvard Medical School from 2007 to 2008. Currently, he is a professor of Institute of Image Processing and Pattern Recognition, Shanghai Jiaotong University. His research interests include data mining, pattern recognition, and bioinformatics. Dr. Shen has published more than 60 papers and constructed 20 bioinformatics severs in these areas and he serves the editorial members of several international journals.

Dong-Jun Yu received the B.S. degree in computer science and the MS degree in artificial intelligence from Jiangsu University of Science and Technology in 1997 and 2000, respectively, and the Ph.D. degree in pattern analysis and machine intelligence from Nanjing University of Science and Technology in 2003. In 2008, he acted as an academic visitor at University of York in UK. He is currently a professor in the School of Computer Science and Engineering of Nanjing University of Science and Technology. His current interests include pattern recognition, data mining and bioinformatics.