2015 IEEE 15th International Conference on Data Mining Workshops

Accurate classification of biological data using ensembles

Manju Bhardwaj
Department of Computer Science
Maitreyi College
Delhi University, Delhi 110007, India
Email: mbhardwaj@maitreyi.du.ac.in

Debasis Dash
GN Ramachandran Knowledge
Centre for Genome Informatics,
CSIR-Institute of Genomics
and Integrative Biology,
Mall Road,
Delhi 110007, India
Email: ddash@igib.res.in

Vasudha Bhatnagar
Department of Computer Science
Delhi University, Delhi 110007, India
Email: vbhatnagar@cs.du.ac.in

***Abstract*—Predicting the class to which a given protein sequence belongs is a challenging research area in bioinformatics. Machine learning techniques have been successfully applied to protein prediction problems like allergen prediction, mitochondrial prediction and toxin prediction. Physicochemical properties derived from sequences of amino acids have been commonly used for this purpose.**

In this paper, we propose an SVM based ensemble method for classification of protein datasets. The constituent classifiers of the ensemble are generated in a sequential manner, each one attempting to rectify mistakes made by previous one. The ensemble is aptly called Self-Chastisting Ensemble (SCE) because of the iterative refinement each classifier carries out over the previous one. We present two versions of the algorithm: SCE-Bal for balanced datasets and SCE-Imbal for imbalanced datasets. Empirical results further demonstrate that the algorithm delivers superior performance using simple and computationally efficient features (amino acid composition and dipeptide composition) compared to other machine learning methods using complex feature sets.

I. Introduction

Classification of proteins based on various cellular functions is the key to understand biology of an organism [1]. For example, classification of proteins on the basis of their chaperone dependence for aggregation [2] can provide valuable insights into the management of aggregation prone proteome by eubacterial organisms during their evolution. Researchers have employed variety of computational techniques ranging from simple homology based algorithms to complex machine learning based classifiers, to solve the problem of protein function prediction. Intense efforts by bioinformatics researchers during last two decades have resulted in several classification models for protein classification [3], [4], [5], [6], [7], [8].

Machine learning techniques like decision trees [9], random forests [9], position specific scoring matrix [8], artificial neural networks [10], naive bayes [11], [9], hidden markov models [12] and hybrid methods [6], [5] have been extensively been used for developing prediction models in proteomics. Interestingly, Jensen and Bateman pointed out in their editorial [13] that since 1970, artificial neural networks has been the most commonly used technique for solving problems in bioinformatics.

Recent years have witnessed a distinct surge in the popularity of Support Vector Machines (SVM) as an accurate, efficient and robust technique for supervised learning [11], [14], [4], [3], [6]. The swing in favor of SVM can be attributed to its strong theoretical foundations and a high generalization capability [15]. There are ample theoretical and empirical works that vindicate this view. However, most practical implementations of SVM deliver approximated solutions to large real life optimization problems. Memory and space constraints pose hindrance in searching the truly optimal solution of the optimization problem. Evidently, there is a gap between the theoretical and practical solutions of optimization problem. Classification ensembles with SVM as base classifier have been proposed in an attempt to fill this gap leading to performance improvement of support vector machine [16], [17].

A. SVM-based Ensembles

Study of classification ensemble methods reveals the low popularity SVM suffers as the choice of base classifier. Weak classifiers such as decision tree and neural networks are highly preferred as base classifiers in classification ensembles. Ironically, the high quality performance of SVM is the primary reason for this aversion. SVM enthusiasts have created SVM ensembles by using the well known concepts of bagging and boosting [16].

In this work, an SVM ensemble is introduced which refines itself in multiple iterations. Each classifier is trained on a judicious mix of training data and errors made by previous classifier on training data. Preliminary results on this strategy have been encouraging [18], [19] and motivated us to further improve it for handling idiosyncrasies of protein data.

B. Challenges in protein prediction

Modelling of biological processes poses formidable challenge to scientists because biological events (at both micro and macro level) do not follow any known mathematical principles [20]. Stochastic behavior of biological events, a well

978-1-4673-8493-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ICDMW.2015.229

accepted fact, has motivated application of machine learning techniques for prediction of various types of cellular behavior in the process [21], [22].

Next challenge in prediction of protein types results from the fact that all data available for analysis is obtained from experimental observations. Human and machine errors at various levels introduce noise in data [23].

Many real life protein datasets suffer from the problem of class imbalance, which thwarts supervised learning process [24], [25]. In such data sets examples of positive (minority) class are significantly less than those of negative (majority) class leading to severe class imbalance. Constructing high quality classifiers for such imbalanced training data sets is one of the major challenges in machine learning, since traditional classification algorithms tend to get biased towards majority class.

Given a protein data sequence, a large number of structural and physiochemical features can be generated [26]. Features like amino acid composition [6], dipeptide composition [6], pseudoamino acid composition [3] are commonly used in protein sequence classification. The choice of feature set which identifies a class of proteins distinctively is another important issue [27].

C. Research proposal

In this paper, we propose design of an SVM ensemble that consists of dependant classifiers. Each classifier improves the performance on the training set from the previous one by focussing on the training samples that have been classified erroneously.

Constructing an ensemble of weakened SVM classifiers[1] exploits the opportunity for improved supervised learning in space of SVM ensembles. The proposal is particularly suitable for classification of biological data because of the following reasons:

1) It uses an acclaimed machine learning technique (SVM) for construction of predictive model in absence of any known theoretical model for protein classification.
2) The base classifier (SVM) is able to handle noisy data prevalent in biological sciences, and hence is an apt choice.
3) A variation in the proposal is specially suited for imbalanced data sets that are common in the bioinformatics domain.
4) The method is powerful enough to yield improved results on tested datasets using simple and computationally cheap features in contrast to physico-chemical features used by other algorithms.

We present a proposal for designing a classification ensemble using probabilistic SVM, where component models are dependant. Each model focuses on the training errors made by the previous model, in an attempt to improve performance on training set. In a sense, each model "chastises" itself because of the errors made by its immediate ancestor model. The ensemble consists of all the models thus created. We propose two versions of this algorithm - *SCE-Bal* for balanced datasets and *SCE-Imbal* for imbalanced datasets. We further demonstrate in this paper that amino acid composition and dipeptide composition can be successfully used to obtain better results than other methods which use complex feature sets.

It is hard to miss the uncanny similarity with Adaboost [29] algorithm. Adaboost maintains a set of weights over the original training set and increases the weights of incorrectly classified instances while decreasing the weights of correctly classified instances. SCE picks up the idea of emphasizing on "mistakes" from Adaboost. The choice of points to be included in next training set is determined by adaboost assigned weights whereas SCE selects the mistakes from the last iteration only. Another point of difference is that SCE does not associate weights with instances, it undersamples the mistakes by using the probability values returned by probabilistic SVM.

D. Organization of the Paper

The rest of the paper is organized as follows. Section II presents the related work with respect to classification of proteins and use of SVM ensembles in machine learning and bioinformatics. SCE algorithm for balanced as well as imbalanced data is presented in Section III. Section IV briefly describes the results obtained on application of SCE to selected protein datasets. Finally, Section V concludes the paper.

II. Related Work

A. Protein Classification

A large variety of protein classification problems have been solved using the primary structure of given sequences. Features like amino acid composition (AAC), dipeptide composition (DPC) and pseudoamino acid composition (PseAAC) have been commonly used for protein classification problems.

Kandaswamy et. al [30] propose random forest approach for prediction of "antifreeze proteins" based on 119 sequence derived features like physiochemical properties and secondary structure (helix/strand/coil). SVM has also been applied on AAC and DPC feature sets for identification of allergens [6], DNA-binding proteins [4], fungal adhesins [7], secretory proteins of malaria parasite [8] and mitochondrial proteins [5]. [5] also uses split AAC for mitochondrial prediction. Gao et al. [31] propose a nuclear receptor sub-families predicting method using PseAAC as the feature expression. PseAAC has also been used for effective prediction of antiviral peptides [32] using Adaboost algorithm with five different base classifiers, i.e., RBF, naive bayes, J48, REPTree and decision stump. SVM with a local alignment kernel, applied on PseAAC has been effectively used for prediction of anti-cancer peptides [33].

B. SVM Ensembles in Machine Learning

Ensemble learning is an effective technique that has increasingly been adopted to combine multiple prediction models to improve overall predictive performance [34]. The three most popular ensemble methods are bagging [35], boosting [29] and random forests [36].

[1] In general weak classifiers are preferred for ensemble construction [28].

SVM ensembles have been extensively used by machine learning community for applications like image classification [37], scene classification [38], face recognition [39], [40] and pattern recognition problems like handwritten digit recognition and fraud detection [17].

C. SVM ensembles in bioinformatics

Computational biology has witnessed an increased use of ensemble methods because of their ability to efficiently handle small sample sizes, high-dimensionality and complex data structures [41]. SVM ensembles are the upcoming promising direction in ensemble methods in bioinformatics.

Bagged ensembles using SVM as the base classifier have been used for glycosylation site prediction [42] and gene function recognition [43]. Multiple SVM models each with a different kernel are combined to create an SVM ensemble for prediction of transcription start sites [44].

D. Classification in imbalanced datasets

Classification of imbalanced datasets has been extensively studied during last decade and effective methods have been designed which perform well on majority as well as minority class. The problem has been addressed for designing SVM ensembles by effective use of undersampling, oversampling and cost sensitive methods [45], [46], [47], [48].

III. Algorithm

SCE is an SVM based ensemble method, where multiple prediction models are generated by amending the training set in each iteration. The first model in the ensemble is trained on the full training set and the same is predicted by the model. All subsequent models are trained on the training dataset curated as follows. The mistakes made by the previous model on full training set along with the unbounded support vectors accumulated from all previous models form the new training dataset. In case the mistakes form a larger part of training set, mistakes equal to the number of accumulated unbounded support vectors are selected. The selected mistakes are the ones which have highest probability of being misclassified. Further, 50% positive and 50% negative mistakes are selected to have an equal representation from both the classes. The ensemble "chastises" or refines itself by emphasizing on mistakes currently made in prediction. To balance the mistakes, each training set has a representation of correctly classified instances in the form of unbounded support vectors generated from previous models.

A. SCE-Bal: SCE for Balanced datasets

This algorithm can be used for datasets where there is either equal or slight imbalance in representation of the two classes. Algorithm 1 presents an outline of SCE-Bal algorithm.

The original training set ($Train$) is used as the first training set T (Step 1). Ensemble E is initialized to null (Step 2). Set S, which holds the unbounded support vectors from all generated models, is initialized to null (Step 3).

For each dataset, num iterations are carried out (Steps 4–22). In each iteration, a probabilistic SVM model M is generated using training set T (Step 4). The model M is then used to predict the training set $Train$ (Step 5). The incorrectly classified instances of $Train$ are stored in $\mathcal{M}$ (Step 6). If M does not make any mistakes on $Train$, the model generation process terminates (Steps 7–9). If the accuracy attained by the model M on $Train$ exceeds the user specified $cutoff$ value, the current model is included in ensemble E and unbounded support vectors in M are added to the set S (Steps 11–13). If the number of mistakes is greater than number of instances in S (ns), the algorithm retains only ns mistakes in $\mathcal{M}$ – $ns/2$ positive mistakes and $ns/2$ negative mistakes which show a higher probability of misclassification (Steps 19–20). The next training set (T) is then generated by concatenating S with the set of mistakes ($\mathcal{M}$) (Step 22). The algorithm then moves on to the next iteration.

If an even number of classifiers is generated, the last model is discarded since majority voting is to be used to combine outputs of obtained classifiers (Steps 23–25).

Algorithm 1 SCE-Bal: SCE for balanced datasets

Require: $Train$: Training dataset, $Test$: Test dataset, num: Ensemble size, $cutoff$: Accuracy cutoff value used for model selection

1: T = $Train$
2: $E = \phi$; {E is the ensemble initialized to null}
3: $S = \phi$ {S is the set of unbounded support vectors initialized to null}
4: **while** ($|E| \leq num$) **do**
5: Generate SVM model M on T with acc as accuracy reported on $Train$
6: Store the incorrectly predicted instances (mistakes) of $Train$ in $\mathcal{M}$
7: **if** ($\mathcal{M} = \phi$) **then**
8: ExitLoop
9: **end if**
10: **if** ($acc \geq cutoff$) **then**
11: Add unbounded Support Vectors in M to S
12: Include M in E
13: **else**
14: Discard M
15: **end if**
16: Let ns be the number of unique instances in S
17: **if** ($|\mathcal{M}| > ns$) **then**
18: Sort the mistakes in $\mathcal{M}$ in decreasing order of probability of misclassification
19: Retain top $ns/2$ positive mistakes and top $ns/2$ negative mistakes in $\mathcal{M}$
20: **end if**
21: Create the next version of training set T by concatenating S and $\mathcal{M}$
22: **end while**
23: **if** ($|E|$ % 2 = 0) **then**
24: Remove the last classifier from ensemble E
25: **end if**

B. SCE-Imbal: SCE for imbalanced datasets

Many real life biological datasets are often imbalanced, e.g., number of non-allergen proteins is very large as compared to known allergens. In such cases, normal classifiers tend to learn more from the larger class and hence, get biased towards it. To handle this issue, a new version of SCE, namely, SCE-Imbal is proposed in this section.

SCE-Imbal is suitable for datasets where there is a large disparity in the number of instances belonging to the two classes. The class with a smaller representation in the dataset is usually called positive or minority class whereas the other class is majority or negative class. From this point onwards, the two classes will be referred to as positive and negative classes respectively. Algorithm 2 presents an outline of SCE algorithm for imbalanced datasets.

The basic idea of the algorithm is same as SCE-Bal but it focusses more on the smaller positive class. In the first balanced training set T, all instances of positive class are always included. To balance T, an equal number of negative class instances are also included in T. Similarly, if the mistakes are to be undersampled, all positive mistakes are selected for inclusion in training set. The remaining mistakes picked up from negative class are the ones with higher probability of misclassification.

Since the data is imbalanced, accuracy is not a suitable performance measure for model selection (Sec. IV-A). The user can tune the ensemble training by use of any suitable parameter like precision, recall or gmean. However, the method is general and can accommodate other user performance measures for model selection.

The basic working of SCE-Imbal is quite similar to SCE-Bal except for the following:

1) SCE-Bal begins with $Train$ as the first training set but SCE-Imbal begins with selects a balanced subset of $Train$ as the first training set T. All positive class instances and equal number of randomly sampled negative class instances constitute T (Step 3).
2) SCE-Bal uses accuracy as a criterion for model selection because both the classes are equally represented but SCE-Imbal uses parameters like precision, recall or gmean as a criterion for model selection (Step 11).
3) If number of mistakes exceeds the number of instances in S, SCE-Bal selects an equal number of positive as well as negative mistakes but SCE-Imbal selects all positive and required number of negative mistakes (Step 19).

C. Intuition

The key idea in design of SCE is the use of "mistakes" to improve the prediction model iteratively. The set of mistakes encountered on the complete training set is assimilated in the next version of training set, as the algorithm strives to correctly classify these mistakes in the next iteration. These mistakes are expected to maneuver the separating hyperplane to a favorable orientation so that rectification takes place to the best possible extent.

Algorithm 2 SCE-Imbal: SCE for imbalanced datasets

Require: $Train$: Training dataset, $Test$: Test dataset, num: Ensemble size, msr: Performance measure used for model selection, $cutoff$: Performance measure's cutoff value

1: $E = \phi$; {E is the ensemble initialized to null}
2: $S = \phi$ {S is the set of unbounded support vectors initialized to null}
3: T=All positive class instances from $Train$ + an equal number of randomly selected negative class instances from $Train$
4: **while** ($|E| \leq num$) **do**
5: Generate SVM model M on T with val reported as value of msr on Train
6: Store the incorrectly predicted instances (mistakes) of $Train$ in $\mathcal{M}$
7: **if** ($\mathcal{M} = \phi$) **then**
8: ExitLoop
9: **end if**
10: **if** ($val \geq cutoff$) **then**
11: Add unbounded Support Vectors in M to S
12: Include M in E
13: **else**
14: Discard M
15: **end if**
16: Let ns be the number of unique instances in S
17: **if** ($|\mathcal{M}| > ns$) **then**
18: Sort the negative mistakes in $\mathcal{M}$ in decreasing order of probability of misclassification
19: Retain only ns mistakes in $\mathcal{M}$-all positive mistakes and top negative mistakes
20: **end if**
21: Create the next training set T by concatenating S and $\mathcal{M}$
22: **end while**
23: **if** ($|E|$ % 2 = 0) **then**
24: Remove the last classifier from E
25: **end if**

The set of unbounded support vectors play a sub-cardinal role in the maneuver. Lying on margins of the separating hyperplane, they are representatives of the margins of the two classes in the considered training set[2]. The set of unbounded support vectors accumulated over the iterations intuitively defines the contours of the class boundaries in the original training set.

The algorithm maintains a balance within each generated training set as described below. All iterations after the first one, train the SVM on a set that is balanced w.r.t. correctly classified instances (unbounded support vectors) and errors. In each subsequent iteration, the balance in training set is preserved by pooling in equal number of mistakes and unbounded support vectors. Sometimes, the model may throw

[2]The bounded support vectors may lie beyond the margins and sometimes, on the wrong side of margin also.

up more mistakes than the accumulated unbounded support vectors. In such situations, the mistakes are undersampled so that they do not overwhelm the separating hyperplane. The mistakes farthest from the separating hyperplane are favored while undersampling so as to enable the generated machine to correctly classify the "hard" mistakes. Use of probabilistic SVM aids in this task.

Strategy for undersampling in balanced and imbalanced datasets is different due to inherent difference in the class distributions. In balanced datasets, equal representation of both classes advocates selection of equal number of mistakes from positive as well as negative class. In case of imbalanced datasets, positive mistakes are preferred and negative mistakes are included only in case of deficiency. Preference for positive mistakes enables the machine to focus more on the positive class.

In the case of imbalanced datasets, the initial training set is balanced to hold equal number of instances from each class, so that the first SVM model and hence, the subsequent models are not bad performers for positive class.

IV. Experimental section

This section presents the results obtained on application of SCE to selected protein datasets. The choice of datasets is limited to those which were available on the corresponding web servers and the results were also readily available for comparison. The datasets are available in the form of protein sequences in fasta format. Given the dataset, each sequence is converted to a fixed length feature vector, the features used in this work are described in Section IV-B. All the experiments are conducted on Pentium dual-core PC and Pentium quad-core desktops.

A. Performance metrics

Accuracy is the most commonly used performance measure for evaluating a predictive model. But it may not work appropriately for data with imbalanced distribution of classes. For example, if a dataset of 10,000 instances consists of 9,900 instances belonging to class -1 and only 100 instances belonging to class +1, an accuracy of 99% although high, is a deceptive indicator. If all the instances are predicted as belonging to class +1, even then an accuracy of 99% (9900 correct/10000 total) is achieved, although not even a single instance belonging to class +1 has been predicted correctly. Hence, in such cases, metrics like sensitivity, specificity and precision are used as alternative to accuracy. Additionally it Gmean and *MCC* are commonly used for assessing classifier performance for imbalanced datasets. Gmean is used by machine learning community while the bioinformatics community uses Matthew's correlation coefficient (MCC) to measure predictive performance on the two classes taken together.

If a positive (negative) example truly identified as positive (negative) is called TP or true positive (TN or true negative) and a wrongly classified negative is called FN or false negative (FP or false positive), then gmean and MCC can be defined as:

$$gmean = \sqrt{\frac{TP}{TP+FN} * \frac{TN}{TN+FP}} \tag{1}$$

$$MCC = \frac{(TP*TN)-(FP*FN)}{\sqrt{(TP+FN)(TP+FP)(TN+FN)(TN+FP)}} \tag{2}$$

B. Feature sets for Protein Sequences

Proteins are long chains of amino acids joined together. In all, there are 20 amino acids. Information present in amino acid sequences of proteins has been found to be useful for prediction of protein function by machine learning methods. The compositional features, namely, amino acid composition and dipeptide composition [6] are used in this work for solving protein prediction problem:

1) *Amino Acid Composition (AAC)*: It is expressed as a vector of 20 numerical attributes [f_1,f_2, ... f_{20}], where each f_i is the normalized frequency of i^{th} amino acid in the protein sequence of length L given by
$$f_i = \frac{n_i}{L}$$
where n_i is the count of i^{th} amino acid in the protein.
2) *Dipeptide Composition (DPC)*: It is expressed as a vector of 400 numerical attributes [d_{11},d_{12}, ... $d_{20,20}$], where each $d_{i,j}$ is the normalized frequency of i^{th} and j^{th} amino acids appearing in consecutive positions in the protein sequence of length L given by
$$d_{i,j} = \frac{n_{i,j}}{L(L-1)}$$
where $n_{i,j}$ is the count of occurences of pair formed by i^{th} and j^{th} amino acid in the protein.

For example, for an amino acid sequence $AAMARGMKC$, amino acid composition for amino acids A and C is computed as:

$$f_A = \frac{n_A}{L} = \frac{4}{10} = 0.4$$

$$f_C = \frac{n_C}{L} = \frac{1}{10} = 0.1$$

For the same sequence, dipeptide composition for AA, AM and MA is computed as:

$$d_{A,A} = \frac{n_{A,A}}{L*(L-1)} = \frac{2}{10*9} = 0.022$$

$$d_{A,M} = \frac{n_{A,M}}{L*(L-1)} = \frac{1}{10*9} = 0.011$$

$$d_{M,A} = \frac{n_{M,A}}{L*(L-1)} = \frac{1}{10*9} = 0.011$$

It is worth noting that given a sequence of length L, computation of the above mentioned feature sets is O(L) and require a single scan of the sequence.

C. Results

This section presents the experimental evaluation of the two versions of SCE algorithm. We use public protein datasets used by recently published works on protein prediction and compare our results with them. We also compare the performance of SCE with machine learning methods well known for classification of complex datasets [9].

The experimental setup is as follows. LIBSVM [49] implementation is used for generation of base classifiers. RBF kernel is used and parameters are optimized for each dataset. The choice of kernel and optimization of model parameters is motivated by earlier studies [50], [51], [52]. For balanced datasets, the accuracy cutoff of 50% is used for model selection. For imbalanced datasets, "gmean" is used as the performance measure and cutoff value of 33% is used . Weka [53] implementations of LR and MLP is used with default parameters used for both the methods.

SCE, LR and MLP are run for two feature sets: AAC (20 features) and AAC+DPC (420 features)[3] for each selected dataset.

1) AllergenFP dataset [54]: This dataset consists of 2427 allergens and 2427 non-allergens protein sequences and was downloaded from [55]. This dataset was also used for allergen prediction studies in AllerTOP [9] and Allergen-ANN [10]. Since the dataset is perfectly balanced, SCE-Bal was applied to this dataset and 10 fold CV was carried out.

Same set of features is used by the three above mentioned works. Each protein sequence is described by five E-descriptors derived by principal component analysis of 237 physiochemical properties:

1) E1: Hydrophobicity of Amino Acids
2) E2: Size of Amino Acids
3) E3: Their helix forming propensity
4) E4 : Relative abundance of Amino Acids
5) E5: Beta-strand forming propensity

The different length strings of E-descriptors are transformed into equal length vectors by using Auto-covariance and cross-covariance (ACC). The ACC values are then used to generate a unique binary fingerprint for each protein, which consists of 375 units and 3750 nulls.

AllergenFP [54] prediction model is based on Tanimoto coefficient similarity search. AllerTop [9] uses other machine learning methods like logistic regression, decision trees, naive bayes, random forest, multilayer perceptron and k-nearest neighbor (kNN) on the given dataset. Of all the techniques, kNN reports the best results. [10] uses artificial neural networks for classifying the data. Note that accuracy and MCC values are not reported for this method. In each of these works, 10 fold CV results have been reported on the dataset thus making them comparable with SCE-Bal results (Table I).

It can be seen from the table that AAC-based SCE-Bal produces results comparable with AllerTop which is based on a feature set obtained after application of PCA to a set of 237 features. (AAC+DPC)-based SCE-Bal attains best performance metrics amongst all techniques and feature sets. Though the number of features used in AAC+DPC feature set is highest, by no means it is suggestive of an overhead for the proposed protein prediction method. Aller-family of protein predictors uses PCA to reduce the number of features which is an added computational expense over 237 features computed for each sequence. We can therefore justify the superiority of SCE-Bal over other methods.

[3]MLP was unable to produce results for (AAP+DPC) feature set for larger datasets, i.e., AllergenFP and AFP-Pred.

TABLE I
COMPARING ALLERGENFP [54], ALLERTOP [9], ALLERGEN-ANN [10], LR AND MLP WITH SCE-BAL; ACC-ACCURACY, SPEC-SPECIFICITY, SEN-SENSITIVITY, MCC-MATTHEW'S CORRELATION COEFFICIENT

Predictor	#Ftrs	Acc	Spec	Sen	MCC
AllergenFP	237	85.00	85.00	85.00	0.70
AllerTop	237	88.70	86.70	90.70	0.775
Allergen-ANN	237	-	82.00	82.00	-
LR(AAC)	20	77.51	70.86	82.97	0.54
MLP(AAC)	20	81.54	76.76	85.55	0.63
SCE-Bal(AAC)	20	89.06	**87.23**	90.57	0.78
LR(AAC+DPC)	420	83.81	80.74	86.33	0.67
SCE-Bal(AAC+DPC)	420	**90.72**	87.10	**93.69**	**0.81**

TABLE II
COMPARING ALGPRED [6], CHOU-PSEAAC [14], LR AND MLP WITH SCE-BAL; ACC-ACCURACY, SPEC-SPECIFICITY, SEN-SENSITIVITY, MCC-MATTHEW'S CORRELATION COEFFICIENT

Predictor	#Ftrs	Acc	Spec	Sen	MCC
AlgPred	20	85.02	88.87	81.86	0.70
Chou-PseAAC	2050	91.20	89.18	92.86	0.82
LR(AAC)	20	84.93	84.49	85.29	0.70
MLP(AAC)	20	85.79	85.02	86.43	0.71
SCE-Bal(AAC)	20	91.65	89.37	93.53	0.83
LR(AAC+DPC)	420	74.73	72.47	76.57	0.49
SCE-Bal(AAC+DPC)	420	**92.54**	**90.41**	**94.29**	**0.85**

2) AlgPred dataset [6]: This allergen dataset consists of 578 allergen and 700 non-allergen protein sequences and is available at [56]. This dataset was referenced in AlgPred [6] and Chou's pseudoamino acid composition based predictor (Chou-PseAAC) [14]. 5 fold CV results have been reported in both the papers. We follow the same strategy for experimentation using SCE-Bal.

Authors of AlgPred established that among several feature sets and machine learning methods, SVM with AAC feature set performs best. Authors of [14] compute the pseudoAmino acid composition based on six physiochemical properties– hydrophobicity, hydrophilicity, side-chain mass, pk of alpha-COOH group, pk of alpha-NH3 group and pI at 25 C, to form 41 different combinations. For each of the combinations, 50 features are generated thus generating a total of 2050 features.

Table II reports results by LR, MLP and SCE-Bal along with the best results obtained by the referenced predictors. It can be seen from the table that AAC-based SCE-Bal yields better results than AlgPred and only marginally better than Chou-PseAAC predictor. (AAC+DPC)-based SCE-Bal performs best with the added advantage of smaller feature set over Chou-

TABLE III
COMPARING AFP-PRED [30], LR AND MLP WITH SCE-IMBAL; ACC-ACCURACY, SPEC-SPECIFICITY, SEN-SENSITIVITY, MCC-MATTHEW'S CORRELATION COEFFICIENT

Predictor	#Ftrs	Acc	Spec	Sen	MCC
AFP-Pred	119	83.38	**84.67**	82.32	0.67
LR(AAC)	20	95.65	26.72	99.37	0.40
MLP(AAC)	20	96.59	46.76	99.20	0.58
SCE-Imbal(AAC)	20	97.14	58.67	99.16	0.67
LR(AAC+DPC)	420	95.79	53.86	97.99	0.54
SCE-Imbal(AAC+DPC)	420	**97.78**	63.33	**99.58**	**0.74**

PseAAC predictor.

D. AFPPred dataset [30]

This dataset used in AFP-Pred [30] consists of 481 antifreeze and 9493 non-antifreeze protein sequences. SCE-Imbal was used for this dataset.

AFP-Pred [30] uses random forest method for the prediction of antifreeze proteins from protein sequences using 119 features. Heterogeneous features (frequencies of functional groups, frequencies of short peptides, physico chemical properties etc.) derived from sequences, have been used. Five fold CV results have been reported in the paper.

Table III shows that AFP-Pred reports higher specificity. However, SCE-Imbal is a better performer in terms of sensitivity, accuracy and MCC. Another interesting observation is the low specificity values for LR and MLP indicating poor recognition of positive (smaller) class.

V. CONCLUSION

Machine learning methods like random forests, decision tress and naive bayes method have been successfully used by bioinformatics community for solving protein prediction problem. In this work, we propose an SVM based Self Chastising Ensemble (SCE) in which each constituent model tends to correct errors made by the previous model, thus gradually refining the ensemble. Two versions of SCE have been proposed – SCE-Bal for balanced data sets and SCE-Imbal for imbalanced datasets.

Use of simple compositional features for protein function prediction by SCE yields better performance than some well known earlier works.

VI. FUTURE WORK

Current study is based on intuition and empirical evidence, the theoretical underpinnings need to be investigated. It would also be interesting to study the performance of SCE on simulated datasets with outliers. Further, this method can be enhanced to handle multiclass datasets.

REFERENCES

[1] C. Brun, F. Chevenet, D. Martin, J. Wojcik, A. Guénoche, B. Jacq *et al.*, "Functional classification of proteins for the prediction of cellular function from a protein-protein interaction network," *Genome biology*, vol. 5, no. 1, pp. R6–R6, 2004.

[2] R. D. Roy, M. Bhardwaj, V. Bhatnagar, K. Chakraborty, and D. Dash, "How do eubacterial organisms manage aggregation-prone proteome?" *F1000Research*, vol. 3, 2014.

[3] H. Mohbatkar, "Prediction of cyclin proteins using chou's pseudo amino acid composition," *Protein and Peptide letters*, vol. 10, no. 17, pp. 1207–1214, 2010.

[4] M. Kumar, M. Gromiha, and G. Raghava, "Identification of dna-binding proteins using support vector machines and evolutionary profiles," *BMC Bioinformatics*, vol. 8:463, 2007.

[5] M. Kumar, R. Verma, and G. P. Raghava, "Prediction of mitochondrial proteins using support vector machine and hidden markov model," *The Journal of Biological chemistry*, vol. 281, no. 9, pp. 5357–5363, 2006.

[6] S. Saha and G. P. Raghava, "Algpred: prediction of allergic proteins and mapping of ige epitopes," *Nucleic Acids Research*, vol. 34, 2006.

[7] J. Ramana and D. Gupta, "Faapred: A svm-based prediction method for fungal adhesins and adhesin-like proteins," *PLOS One*, vol. 5(3), 2010.

[8] R. Verma, A. Tiwari, S. Kaur, G. C. Varshney, and G. P. Raghava, "Identification of proteins secreted by malaria parasite into erythrocyte using svm and pssm profiles," *BMC Bioinformatics*, vol. 9:201, 2008.

[9] I. Dimitrov, I. Bangkov, D. R. Flower, and I. Doytchinova, "Allertop v.2 - a server for in silico prediction of allergens," *J. Mol. Model*, vol. 20, 2014.

[10] I. Dimitrov, L. Naneva, I. bangkov, and I. Doytchinova, "Allergenicity prediction by artificial neural networks," *Chemometrics*, vol. 28, pp. 282–286, 2014.

[11] H. X. Dang and C. B. Lawrence, "Allerdictor: Fast allergen prediction using text classification techniques," *Bioinformatics Advance Access*, 2014.

[12] K. B. Li and et.al, "Predicting allergenic proteins using wavelet transform," *Bioinformatics*, vol. 20, pp. 2572 – 2578, 2004.

[13] L. J. Jensen and A. Bateman, "Editorial: The rise and fall of supervised machine learning techniques," *Bioinformatics*, vol. 27, no. 24, pp. 3331–3332, 2011.

[14] H. Mohabatkar, M. M. Beigi, K. Abdolahi, and S. Mohsenzadeh, "Prediction of allergenic proteins by means of concept of chou's pseudo amino acid composition and a machine learning approach," *Medicinal Chemistry*, vol. 9, pp. 133–137, 2013.

[15] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[16] H.-C. Kim, S. Pang, H.-M. Je, D. Kim, and S. Y. Bang, "Constructing support vector machine ensemble," *Pattern recognition*, vol. 36, no. 12, pp. 2757–2767, 2003.

[17] ——, "Pattern classification using support vector machine ensemble," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2. IEEE, 2002, pp. 160–163.

[18] M. Bhardwaj, T. Gupta, T. Grover, and V. Bhatnagar, "An efficient classifier ensemble using svm," in *Methods and Models in Computer Science, 2009. ICM2CS 2009. Proceeding of International Conference on*. IEEE, 2009, pp. 240–246.

[19] V. Bhatnagar, M. Bhardwaj, and A. Mahabal, "Comparing svm ensembles for imbalanced datasets," in *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*. IEEE, 2010, pp. 651–657.

[20] R. M. May, "Uses and abuses of mathematics in biology," *Science*, vol. 303, no. 5659, pp. 790–793, 2004.

[21] M. W. Libbrecht and W. S. Noble, "Machine learning applications in genetics and genomics," *Nature Reviews Genetics*, 2015.

[22] J. S Bernardes, "A review of protein function prediction under machine learning perspective," *Recent patents on biotechnology*, vol. 7, no. 2, pp. 122–141, 2013.

[23] R. Sloutsky, N. Jimenez, S. J. Swamidass, and K. M. Naegle, "Accounting for noise when clustering biological data," *Briefings in bioinformatics*, vol. 14, no. 4, pp. 423–436, 2013.

[24] R. Kothandan, "Handling class imbalance problem in mirna dataset associated with cancer," *Bioinformation*, vol. 11, no. 1, p. 6, 2015.

[25] W.-J. Lin and J. J. Chen, "Class-imbalanced classifiers for high-dimensional data," *Briefings in bioinformatics*, vol. 14, no. 1, pp. 13–26, 2013.

[26] Z.-R. Li, H. H. Lin, L. Han, L. Jiang, X. Chen, and Y. Z. Chen, "Profeat: a web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence," *Nucleic Acids Research*, vol. 34, no. suppl 2, pp. W32–W37, 2006.

[27] R. D. Roy and D. Dash, "Selection of relevant features from amino acids enables development of robust classifiers," *Amino acids*, vol. 46, no. 5, pp. 1343–1351, 2014.

[28] H. Drucker, "Effect of pruning and early stopping on performance of a boosting ensemble," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 393–406, 2002.

[29] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods," *Annals of statistics*, pp. 1651–1686, 1998.

[30] K. K. Kandaswamy, K.-C. Chou, T. Martinetz, S. Möller, P. Suganthan, S. Sridharan, and G. Pugalenthi, "Afp-pred: A random forest approach for predicting antifreeze proteins from sequence-derived properties," *Journal of Theoretical Biology*, vol. 270, no. 1, pp. 56–62, 2011.

[31] Q. Gaoi, Z. Jin, X. Ye, C. Wu, and J. He, "Prediction of nuclear receptors optimal pseudo amino acid composition," *Analytical Biochemistry*, vol. 387, pp. 54–59, 2008.

[32] M. Zare, H. Mohabatkar, F. K. Faramarzi, M. M. Beigi, and M. Behbahani, "Using chous pseudo amino acid composition and machine learning method to predict the antiviral peptides," *Open Bioinformatics Journal*, vol. 9, pp. 13–19, 2015.

[33] Z. Hajisharifi, M. Piryaiee, M. M. Beigi, M. Behbahani, and H. Mohabatkar, "Predicting anticancer peptides with chou s pseudo amino acid composition and investigating their mutagenicity via ames test," *Journal of Theoretical Biology*, vol. 341, pp. 34–40, 2014.

[34] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple classifier systems*. Springer, 2000, pp. 1–15.

[35] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[36] ——, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[37] B. Linghu and B.-Y. Sun, "Constructing effective svm ensembles for image classification," in *Knowledge Acquisition and Modeling (KAM), 2010 3rd International Symposium on*. IEEE, 2010, pp. 80–83.

[38] R. Yan, Y. Liu, R. Jin, and A. Hauptmann, "On predicting rare classes with svm ensembles in scene classification," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, vol. 3. IEEE, 2003, pp. III–21.

[39] S. Pang, D. Kim, and S. Y. Bang, "Membership authentication in the dynamic group by face classification using svm ensemble," *Pattern Recognition Letters*, vol. 24, no. 1, pp. 215–225, 2003.

[40] R. S. Smith, J. Kittler, M. Hamouz, and J. Illingworth, "Face recognition using angular lda and svm ensembles," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 3. IEEE, 2006, pp. 1008–1012.

[41] P. Yang, Y. Hwa Yang, B. B Zhou, and A. Y Zomaya, "A review of ensemble methods in bioinformatics," *Current Bioinformatics*, vol. 5, no. 4, pp. 296–308, 2010.

[42] C. Caragea, J. Sinapov, A. Silvescu, D. Dobbs, and V. Honavar, "Glycosylation site prediction using ensembles of support vector machine classifiers," *BMC bioinformatics*, vol. 8, no. 1, p. 438, 2007.

[43] Y. Guan, C. L. Myers, D. C. Hess, Z. Barutcuoglu, A. Caudy, and O. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome biology*, vol. 9, no. Suppl 1, p. S3, 2008.

[44] J. J. Gordon, M. W. Towsey, J. M. Hogan, S. A. Mathews, and P. Timms, "Improved prediction of bacterial transcription start sites," *Bioinformatics*, vol. 22, no. 2, pp. 142–148, 2006.

[45] Y. Liu, A. An, and X. Huang, "Boosting prediction accuracy on imbalanced datasets with svm ensembles," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2006, pp. 107–118.

[46] P. Kang and S. Cho, "Eus svms: Ensemble of under-sampled svms for data imbalance problems," in *Neural Information Processing*. Springer, 2006, pp. 837–846.

[47] Y. Liu, X. Yu, J. X. Huang, and A. An, "Combining integrated sampling with svm ensembles for learning from imbalanced datasets," *Information Processing & Management*, vol. 47, no. 4, pp. 617–631, 2011.

[48] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Machine Learning: ECML 2004*. Springer, 2004, pp. 39–50.

[49] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[50] N.-E. Ayat, M. Cheriet, and C. Y. Suen, "Automatic model selection for the optimization of svm kernels," *Pattern Recognition*, vol. 38, no. 10, pp. 1733–1745, 2005.

[51] ——, "Optimization of the svm kernels using an empirical error minimization scheme," in *Pattern Recognition with Support Vector Machines*. Springer, 2002, pp. 354–369.

[52] C. J. Lin, C.-W. Hsu, and C.-C. Chang, "A practical guide to support vector classification," *National Taiwan U., www. csie. ntu. edu. tw/cjlin/papers/guide/guide. pdf*, 2003.

[53] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[54] I. Dimitrov, L. Naneva, I. Doytchinova, and I. Bangov, "Allergenfp: allergenicity prediction by descriptor fingerprints," *Bioinformatics*, p. btt619, 2013.

[55] "AllergenFP web server," http://ddg-pharmfac.net/AllergenFP.

[56] "AlgPred web server," http://www.imtech.res.in/raghava/algpred.