Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 857325, 9 pages
http://dx.doi.org/10.1155/2015/857325

Research Article

Prediction of "Aggregation-Prone" Peptides with Hybrid Classification Approach

Bo Liu,[1] Wenyi Zhang,[2] Longjia Jia,[2] Jianan Wang,[2] Xiaowei Zhao,[2] and Minghao Yin[2]

[1] *School of Physical Education, Northeast Normal University, Changchun 130117, China*
[2] *School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China*

Correspondence should be addressed to Xiaowei Zhao; zhaoxw303@nenu.edu.cn and Minghao Yin; mhyinnenu@163.com

Received 23 July 2014; Revised 29 September 2014; Accepted 29 September 2014

Academic Editor: Shifei Ding

Protein aggregation is a biological phenomenon caused by misfolding proteins aggregation and is associated with a wide variety of diseases, such as Alzheimer's, Parkinson's, and prion diseases. Many studies indicate that protein aggregation is mediated by short "aggregation-prone" peptide segments. Thus, the prediction of aggregation-prone sites plays a crucial role in the research of drug targets. Compared with the labor-intensive and time-consuming experiment approaches, the computational prediction of aggregation-prone sites is much desirable due to their convenience and high efficiency. In this study, we introduce two computational approaches Aggre_Easy and Aggre_Balance for predicting aggregation residues from the sequence information; here, the protein samples are represented by the composition of *k-spaced amino acid pairs* (CKSAAP). And we use the hybrid classification approach to predict aggregation-prone residues, which integrates the naïve Bayes classification to reduce the number of features, and two undersampling approaches EasyEnsemble and BalanceCascade to deal with samples imbalance problem. The Aggre_Easy achieves a promising performance with a sensitivity of 79.47%, a specificity of 80.70% and a MCC of 0.42; the sensitivity, specificity, and MCC of Aggre_Balance reach 70.32%, 80.70% and 0.42. Experimental results show that the performance of Aggre_Easy and Aggre_Balance predictor is better than several other state-of-the-art predictors. A user-friendly web server is built for prediction of aggregation-prone which is freely accessible to public at the website.

1. Introduction

Protein aggregation is a phenomenon caused by misfolding proteins aggregation. Many studies indicate that protein aggregations can cause amyloid fibrils which are associated with a wide variety of diseases, such as Alzheimer's, Parkinson's, and prion diseases [1]. Although the amyloidogenic proteins do not share the homology in sequences or common native fold patterns, they are remarkably similar in β structure [1]. Experiments demonstrate that protein aggregation is mediated by short "aggregation-prone" peptide segments. So the identification of aggregation prone in the protein sequences is the key to finding protein aggregation phenomenon. As we know, traditional experimental identification and characterization of aggregation prone are labor-intensive and expensive. Therefore, the aggregation residues prediction by computational technology attracted more and more attention in the past few years.

Over the past ten years, a large number of computational approaches have been developed to analyze and predict the aggregation prone. Broadly, from the perspective of feature extraction, these approaches can be divided into three categories: experiment-based methods, structure-based methods, and physical-chemical attribute-based methods. For example, Aggrescan [2] proposed by Conchillo-Solé et al. and the saturation mutagenesis analysis [3] performed by López de la Paz and Serrano were both validated by experiments. Among structure-based methods, Galzitskaya et al. [4] used a new parameter, "*mean packing density,*" to detect both amyloidogenic and disordered regions in a protein sequence; SALSA [5], Hexapeptide Conf. Energy [1], and SecStr [6] were proposed by β-sheet structure analysis. NetCSSP [7] developed by Kim et al. used CSSP algorithm and 3D structure to predict the amyloid fibril formation. On the other hand, physical-chemical attribute-based methods such as PaFigure [8] proposed by Tian et al. and Tango [9]

Table 1: The number of aggregation sites and nonaggregation sites in training and testing dataset.

Dataset	Number of the proteins	Number of the aggregation sites	Number of the nonaggregation sites
Training datasets	25	923	5074
Testing datasets	8	335	1499
All datasets	33	1258	6573

developed by Fernandez-Escamilla et al. took into account the physical-chemical principles to predict the aggregation prone. Recently, Tsolis et al. developed two methods named AMYLPRED [10] and AMYLPRED2 [11] which integrated 5 predictors and 11 predictors, respectively.

However, the above methods did not consider that the dataset of aggregation-prone prediction was imbalanced, and some methods were based on the structure information which had high computational complexity. For these reasons, we develop two approaches Aggre_Easy and Aggre_Balance based on the sequence information to predict the aggregation residues. In this study, the protein samples are represented by the composition of *k-spaced amino acid pairs* (CKSAAP) [12–14]. Then, we use a hybrid classification approach to solve sample imbalance problem. The hybrid classification approach integrates the naïve Bayes classification to reduce the number of features and undersampling strategy to deal with the class-imbalance problem. Two undersampling algorithms, EasyEnsemble and BalanceCascade, are both utilized in this paper. The Aggre_Easy achieves a promising performance with a sensitivity of 79.47%, a specificity of 80.70%, and a MCC of 0.42; the sensitivity, specificity, and MCC of Aggre_Balance reach 70.32%, 80.70%, and 0.42. Experimental results show that the performance of Aggre_Easy and Aggre_Balance predictor is better than several other state-of-the-art predictors. A user-friendly web server is built for prediction of aggregation prone which is freely accessible to public at the following website: http://202.198.129.220:8080/AggrePrediction/.

2. Materials and Method

2.1. Datasets. In this paper, we select 33 amyloidogenic proteins to predict "aggregation-prone" peptides. And all the proteins are extracted from Uniprot/Swiss-Prot (Mar, 20, 2013). Moreover, in order to facilitate comparison with the AMYLPRED2, we select the same dataset. For aggregation-prone peptides prediction, 25 proteins are used for training and the remaining 8 proteins for testing. Similar to [11], all experimentally verified aggregation sites in this paper are regarded as positive samples, and the other nonaggregation sites in the same proteins are taken as the negative samples (as can be seen in Supporting Information Text S1 in Supplementary Material available online at http://dx.doi.org/10.1155/2015/857325). The number of proteins samples in each dataset is shown in the Table 1.

We define a possible aggregation-prone peptide $(2\times w+1)$ as the aggregation bond is flanked by "w" residues upstream and "w" residues downstream from the aggregation site. In this paper, we select four different values of w (2, 3, 4, and 5), and the window sizes are the 5, 7, 9, and 11. If the aggregation site is located at the N- or C-terminus of the protein and the length of the peptide is smaller than $(2\times w+1)$, one or multiple "O" characters are added to complement the peptide $(2 \times w + 1)$.

2.2. Protein Encoding Schema. To develop a powerful predictor, an effective mathematical expression to formulate the protein sequences plays an important role, which can truly reflect their intrinsic correlation with the attribute to be predicted [15, 16]. In this research, we use the encoding scheme based the *composition of k-spaced amino acid pairs* (CKSAAP) [12–14], which is successfully used for predicting more types of posttranslational modifications (PTMs) sites (e.g., prediction of palmitoylation sites [13], ubiquitination sites [12], and Phosphorylation Sites [17]). We describe the detailed procedures as follows.

Generally, we define an aggregation prone with a sequence fragment of $(2 \times w + 1)$ amino acids. There are 441 possible amino acid pair types (i.e., $AA, AC, AD, \ldots, OO$). Note that the pairs are extended to the *k-spaced amino acid pairs* (i.e., pairs that are separated by k other amino acids). We can use the vector $(m^k_{AA} m^k_{AC} m^k_{AD} \cdots m^k_{OO})_{441}$ to describe a feature vector. For instance, m^k_{AC} denotes that AC pairs occur m times, and amino acid A is separated by k other amino acid from the amino acid C in the sequences fragment. In this study, based on previous experience, the amino acid pairs for $k = 3, 4, 5$ are jointly considered. So the total dimension of the proposed feature vector is $441 \times (k - 1)$.

2.3. Hybrid Classification Approach. From Section 2.1, we can see that the negative samples are about five times of positive samples, so the traditional learning algorithm, such as SVM, cannot get good performance in this kind of imbalance dataset. For the huge number of features transformed from the CKSAAP encoding, many feature selections methods are carried out to overcome this problem by reducing the dimension of the features. In this paper, we design a hybrid classification approach integrating naïve Bayes classification and two undersampling methods EasyEnsemble and BalanceCascade to predict aggregation sites, which has also been successfully used for text document classification [18, 19]. It takes advantage of both the simplicity of the Bayes technique and the efficient strategy of the undersampling to deal with class-imbalance problem. In Figure 1, the black frame illustrates the process of hybrid classification approach. Firstly, all training proteins or peptides are represented by CKSAAP encoding schema. Secondly, the features of *composition of k-spaced amino acid pairs* are used as the input data for Bayes classification, where the dimensions of process of Bayes classification are based on the number of available classes in the classification task [20]. Finally, we use the undersampling approaches for the predictors. For the query protein, we can use the training model to predict whether or not it is

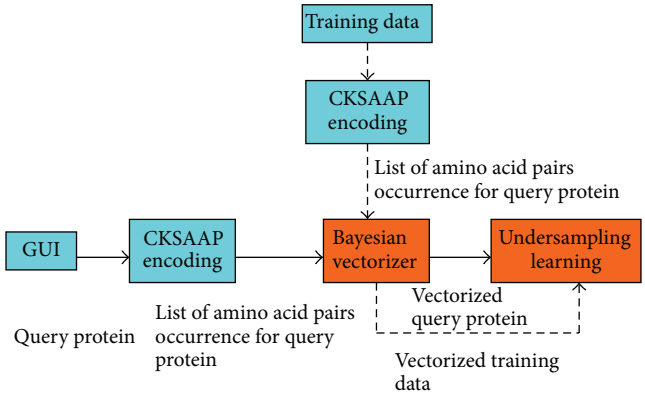

Figure 1: Proposed hybrid classification approach block diagram.

an aggregation protein. The details would be shown in the following sections.

2.3.1. The Bayes Classification Approach. The naïve Bayes classifier [21] starts with the initial step of encoding the sample by extracting the *composition of k-spaced amino acid pairs* (CKSAAP). The list of AAPs (amino acid pairs) is constructed with the assumption that input data contains $\text{aap}_1, \text{aap}_2, \ldots, \text{aap}_{n-1}, \text{aap}_n$, where n is the CKSAAP encoding schema length. And it can be used to create a table; containing the probabilities of the amino acid pair (AAP) in each class. And Table 2 shows the details.

Based on the list of AAP numbers, the trained probabilistic classifier calculates the posterior probability of the particular AAP of the sample being annotated to a particular class by using the formula (1), since each AAP in the input sample contributes to the sample's class probability:

$$\Pr(\text{Class}, \text{AAP}) = \frac{\Pr(\text{AAP}, \text{Class}) \cdot \Pr(\text{Class})}{\Pr(\text{AAP})}. \quad (1)$$

The prior probability, $\Pr(\text{Class}_i)$ can be computed from

$$\Pr(\text{Class}_i) = \frac{\text{Total\_of\_AAPs\_in\_Class}_i}{\text{Total\_of\_AAPs\_in\_Training\_Dataset}}. \quad (2)$$

Meanwhile, we calculate the $\Pr(\text{aap}_j)$, which is represented by the probability of each aap_j in all classes; it is expressed as

$$\Pr(\text{aap}_j) = \frac{\sum \text{numbers\_of\_aap}_i\text{\_in\_all\_Classes}}{\sum \text{numbers\_of\_all\_AAPs\_in\_all\_Classes}}. \quad (3)$$

The total occurrence of a particular AAP in every class can be calculated by searching the training database, which is composed from the lists of AAP occurrences for every class. As previously mentioned, the list of AAP numbers for a class is generated from the analysis of all training samples in the particular class during the initial training stage. The same method can be used to retrieve the sum of numbers of all samples in every class in the training database.

To calculate the likelihood of a particular Class_i with respect to a particular aap_j, the lists of AAP number from the training database is searched to retrieve the numbers of aap_j in Class_i and the sum of all AAPs in Class_i. This information will contribute to the value of $\Pr(\text{aap}_j, \text{Class}_i)$ given in

$$\Pr(\text{aap}_j, \text{Class}_i) = \frac{\text{numbers\_of\_aap}_j\text{\_in\_Class}_i}{\sum \text{numbers\_of\_all\_aaps\_in\_Class}_i}. \quad (4)$$

Based on the derived Bayes' formula for classification and the value of the prior probability Pr(Class), the likelihood Pr(AAP, Class) and evidence Pr(AAP), along with the posterior probability, Pr(Class, AAP) of each AAP in the input data being annotated to each class can be measured.

The probability for an input sample to be annotated to a particular Class_i is calculated by dividing the sum of each of the "Probability" column with the length of the query, n, which is shown in

$$\Pr(\text{Class}_i, \text{Sample}) = \frac{\Pr(\text{Class}_i, \text{aap}_1, \text{aap}_2, \text{aap}_3, \ldots, \text{aap}_{n-1}, \text{aap}_n)}{n}, \quad (5)$$

where $\text{aap}_1, \text{aap}_2, \text{aap}_3, \ldots, \text{aap}_{n-1}, \text{aap}_n$ are the AAPs that are extracted from the input sample.

The Pr(Class, Sample) is the probability value for a sample to be annotated to a class. And if we have class list as Class_1, Class_2, $\text{Class}_3, \ldots, \text{Class}_N$, each sample would have N associated probability values, where sample will have $\Pr(\text{Class}_1, \text{Sample})$, $\Pr(\text{Class}_2, \text{Sample})$, $\Pr(\text{Class}_3, \text{Sample}), \ldots$, and $\Pr(\text{Class}_N, \text{Sample})$. All the probability values of a sample are combined to construct a multidimensional array, which represents the probability distribution of the sample in the vector space. In this way, all the training samples are vectorized into their probability distribution in vector space, in the format of numerical multidimensional arrays, with the number of dimensions depending on the number of classes.

2.3.2. EasyEnsemble and BalanceCascade. Liu et al. proposed EasyEnsemble algorithm and BalanceCascade algorithm [22], which were the undersampling algorithms and were widely used for classification task. EasyEnsemble algorithm extracted several subsets from majority class examples by themselves; for each subset, a classifier was built, and all generated classifiers created an ensemble learning system and then combined them for the final decision by using Adaboost [23]. BalanceCascade algorithm depending on supervised learning methods extracted examples from majority class examples and then created ensemble classifiers with training datasets [24]. The pseudocodes for EasyEnsemble and BalanceCascade were shown in Algorithms 1 and 2.

2.4. Evaluation. In this study, we adopt the 10-fold cross-validation. The dataset is randomly divided into ten equal sets, out of which nine sets are used for training and the remaining one for testing. This procedure is repeated ten times and the final prediction result is the average accuracy of the ten testing sets [25–32].

Four parameters, sensitivity (Sn), specificity (Sp), Q, and Mathew correlation coefficient (MCC), are used to measure

Table 2: Table of AAPs numbers and probabilities.

AAP	Probability Class_1	Probability Class_2	Probability Class_3	$\cdots$	Probability Class_{k-1}	Probability Class_k
aap_1	$\Pr(\text{Class}_1, \text{aap}_1)$	$\Pr(\text{Class}_2, \text{aap}_1)$	$\Pr(\text{Class}_3, \text{aap}_1)$	$\cdots$	$\Pr(\text{Class}_{k-1}, \text{aap}_1)$	$\Pr(\text{Class}_k, \text{aap}_1)$
aap_2	$\Pr(\text{Class}_1, \text{aap}_2)$	$\Pr(\text{Class}_2, \text{aap}_2)$	$\Pr(\text{Class}_3, \text{aap}_2)$	$\cdots$	$\Pr(\text{Class}_{k-1}, \text{aap}_2)$	$\Pr(\text{Class}_k, \text{aap}_2)$
aap_3	$\Pr(\text{Class}_1, \text{aap}_3)$	$\Pr(\text{Class}_2, \text{aap}_3)$	$\Pr(\text{Class}_3, \text{aap}_3)$	$\cdots$	$\Pr(\text{Class}_{k-1}, \text{aap}_3)$	$\Pr(\text{Class}_k, \text{aap}_3)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
aap_{n-1}	$\Pr(\text{Class}_1, \text{aap}_{n-1})$	$\Pr(\text{Class}_2, \text{aap}_{n-1})$	$\Pr(\text{Class}_3, \text{aap}_{n-1})$	$\cdots$	$\Pr(\text{Class}_{k-1}, \text{aap}_{n-1})$	$\Pr(\text{Class}_k, \text{aap}_{n-1})$
aap_n	$\Pr(\text{Class}_1, \text{aap}_n)$	$\Pr(\text{Class}_2, \text{aap}_n)$	$\Pr(\text{Class}_3, \text{aap}_n)$	$\cdots$	$\Pr(\text{Class}_{k-1}, \text{aap}_n)$	$\Pr(\text{Class}_k, \text{aap}_n)$
Total	$\sum \Pr(\text{Class}_1, \text{AAP})$	$\sum \Pr(\text{Class}_2, \text{AAP})$	$\sum \Pr(\text{Class}_3, \text{AAP})$	$\cdots$	$\sum \Pr(\text{Class}_{k-1}, \text{AAP})$	$\sum \Pr(\text{Class}_k, \text{AAP})$
Probability of input sample	$\sum \Pr(\text{Class}_1, \text{AAP})/n$	$\sum \Pr(\text{Class}_2, \text{AAP})/n$	$\sum \Pr(\text{Class}_3, \text{AAP})/n$	$\cdots$	$\sum \Pr(\text{Class}_{k-1}, \text{AAP})/n$	$\sum \Pr(\text{Class}_k, \text{AAP})/n$

Input: Training dataset $S_r = \{(x, y)\}$, the number of individuals T, the number of iterations S_i
(1) Begin
(2) For $i = 1 : T$
(3) Creating a subset S_i^- from negative dataset of S_r^- by using Bootstrap sampling technique, and the number S_i^- is equal to the S_r^+
(4) Use the Adaboost with the weak classifiers $h_{i,j}$ and corresponding weights $\alpha_{i,j}$ to train the individual model N_i, the ensemble's threshold is θ_i, i.e.

$$N_i(x) = \operatorname{sgn}\left(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \theta_i\right).$$

(5) End For
(6) Output: An ensemble like:

$$N(x) = \operatorname{sgn}\left(\sum_{i=1}^{T}\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \sum_{i=1}^{T} \theta_i\right)$$

(7) End

Algorithm 1: The EasyEnsemble algorithm.

Input: Training dataset $S_r = \{(x, y)\}$, the number of individuals T, the number of iterations S_i
(1) Begin
(2) $f = \sqrt[T-1]{|P|/|N|}$ f is the false positive rate (the error rate of misclassifying a majority class example to the minority class) that N_i should achieve
(3) For $I = 1 : T$
(4) Creating a subset S_i^- from negative dataset of S_r^- by using Bootstrap sampling technique, and the number S_i^- is equal to the S_r^+
(5) Use the Adaboost with the weak classifiers $h_{i,j}$ and corresponding weights $\alpha_{i,j}$ to train the individual model N_i, the ensemble's threshold is θ_i, i.e.

$$N_i(x) = \operatorname{sgn}\left(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \theta_i\right)$$

(6) Adjust θ_i such that N_i's false positive rate is f.
(7) Remove from S_r^- all examples that are correctly classified by N_i
(8) End for
(9) Output: A single ensemble like:

$$N(x) = \operatorname{sgn}\left(\sum_{i=1}^{T}\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \sum_{i=1}^{T} \theta_i\right)$$

(10) End

Algorithm 2: The BalanceCascade algorithm.

the performance of our model. They are defined by the following formulas:

$$\mathrm{Sn} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}},$$

$$\mathrm{Sp} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}},$$

$$Q = \frac{\mathrm{Sn} + \mathrm{Sp}}{2},$$

$$\mathrm{MCC} = \frac{(\mathrm{TP} \times \mathrm{TN}) - (\mathrm{FN} \times \mathrm{FP})}{\sqrt{(\mathrm{TP} + \mathrm{FN}) \times (\mathrm{TN} + \mathrm{FP}) \times (\mathrm{TP} + \mathrm{FP}) \times (\mathrm{TN} + \mathrm{FN})}}, \quad (6)$$

where TP, TN, FP, and FN denote the number of true positive, true negative, false positive, and false negative, respectively. For a given dataset, all these values can be obtained from the decision function with fixed cutoff [33–37].

3. Result and Discussion

3.1. The Performance in the Testing Dataset. In this research, we select 33 amyloidogenic proteins for the prediction of "aggregation-prone" peptides. And 25 amyloidogenic proteins are selected for training; there are 923 positive samples and 5074 negative samples; and the rest of 8 amyloidogenic proteins are selected for testing; thus, there are 335 positive samples and 1499 negative samples. The details are shown in Table 1. We define a possible aggregation-prone peptide $(2 \times w + 1)$ as the aggregation bond; "w" is 3, 4, and 5, and the window size is 7, 9, and 11. Next, we use the encoding scheme based on the *composition of k-spaced amino acid pairs* (CKSAAP) to formulate the aggregation-prone peptide, and the "k" is 3, 4, and 5. In Tables 3 and 4, we compare the values of MCC to determine the best values of w and k.

Table 3: The performance for EasyEnsemble learning algorithm in testing dataset.

Window size	The value of the k	Q	MCC	TP	FN	TN	FP
5	3	0.5318	0.0514	136	199	986	513
7	3	0.5438	0.0725	131	204	1044	455
9	3	0.5426	0.0706	130	205	1045	454
11	3	0.5457	0.0779	122	213	1092	407
5	4	0.5443	0.0720	140	195	1006	493
7	**4**	**0.5497**	**0.0827**	**133**	**202**	**1052**	**447**
9	4	0.5347	0.0595	115	220	1090	409
11	4	0.5347	0.0598	112	223	1101	398
5	5	0.5417	0.0677	139	196	1002	497
7	5	0.5188	0.0313	117	218	1034	465
9	5	0.5303	0.0512	116	219	1069	430
11	5	0.5308	0.0542	104	231	1128	371

Table 4: The performance for BalanceCascade learning algorithm in testing dataset.

Window size	The value of the k	Q	MCC	TP	FN	TN	FP
5	3	0.5292	0.0506	107	228	1107	392
7	3	0.5382	0.0694	100	235	1166	333
9	3	0.5197	0.0660	104	231	1142	356
11	3	0.5357	0.0641	101	234	1152	347
5	4	0.5366	0.0649	106	229	1135	367
7	**4**	**0.5405**	**0.0738**	**101**	**234**	**1168**	**331**
9	4	0.5319	0.0580	97	238	1163	336
11	4	0.5253	0.0489	81	254	1213	286
5	5	0.5379	0.0667	108	227	1130	369
7	5	0.5197	0.0353	94	241	1139	360
9	5	0.5293	0.0543	92	243	1175	324
11	5	0.5253	0.0470	81	254	1208	291

We use the hybrid classification approach (naïve Bayes vectorizer and two undersampling algorithms called EasyEnsemble and BalanceCascade) to improve the classification accuracy and performance in the imbalance dataset. For the EasyEnsemble approach, the CART is used to train weak classifiers; the number subset T is 4; the number of iterations S_i is 10 in the each Adaboost ensemble method; the same parameters are used for the BalanceCascade approach. Meanwhile, we perform a 10-fold stratified cross validation. Within each fold, the classification method is repeated 10 times considering that the sampling of subsets introduces randomness. The whole cross validation process is repeated 10 times, and the averages of these 10 cross validations are the final performance of the method.

The average performance of the different parameter is summarized in Tables 3 and 4. When the window size is 7 and the k value was 4, the value of MCC is the highest, 0.0827 for EasyEnsemble learning algorithm and 0.0738 for BalanceCascade learning algorithm in the testing dataset. Thus, we select 7 (window size) and 4 (the k value) as the final parameters of classifier, which is used to comprise with other predictors by 10-fold cross validation in all datasets.

The average Sn of the EasyEnsemble learning algorithm and BalanceCascade learning algorithm is shown in Figures 2 and 4. When the window size is smaller, the value of Sn is higher; for example, window size is 5 and 7, and the Sn is about 0.39~0.41 for EasyEnsemble and 0.27~0.32 for BalanceCascade; on the contrary, when the window size is 9 and 11, the Sn is about 0.34~0.38 for EasyEnsemble and 0.24~0.31 for BalanceCascade. Also in Figures 3 and 5, the average Sp of the EasyEnsemble learning algorithm and BalanceCascade learning algorithm is summarized. It is about 0.66~0.70 for EasyEnsemble and 0.73~0.77 for BalanceCascade, when the window size is 5 and 7; however, it is about 0.69~0.75 for EasyEnsemble and 0.76~0.80 for BalanceCascade, when the window size is 9 and 11. It indicates that smaller window size would be beneficial to predict positive sample; also, the larger the window size is, the more redundant the information is. What's more, the value of Sn is higher for EasyEnsemble than for BalanceCascade, about 10% higher; however, the value of Sp is lower for EasyEnsemble than for BalanceCascade, about 7% lower; it illustrates that the EasyEnsemble would improve the prediction performance of sensitivity, and BalanceCascade would improve the prediction performance of specificity.

3.2. Comparison with Other Predictors. As the result in Table 5, the prediction sensitivity and MCC of Aggre_Easy

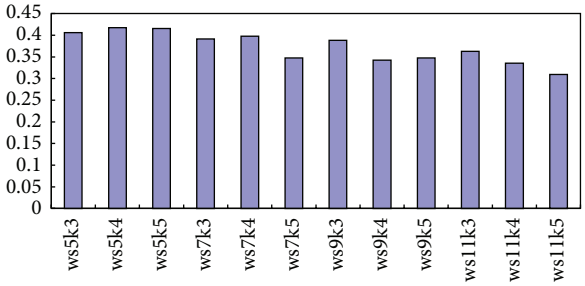

Figure 2: The value of Sn for EasyEnsemble learning algorithm in testing dataset.

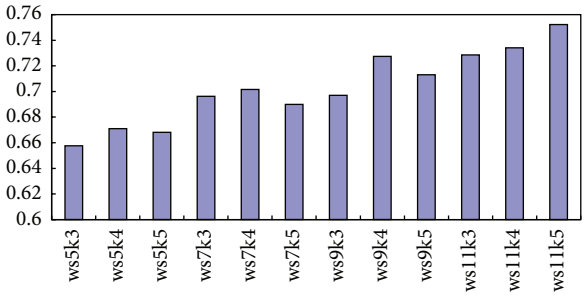

Figure 3: The value of Sp for EasyEnsemble learning algorithm in testing dataset.

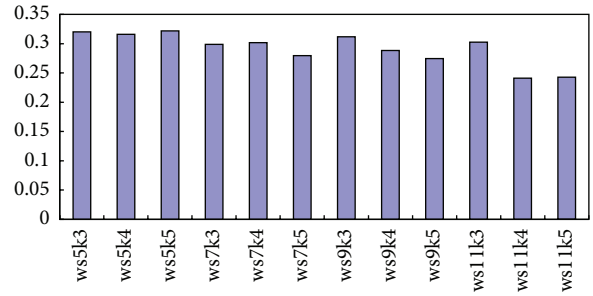

Figure 4: The value of Sn for BalanceCascade learning algorithm in testing dataset.

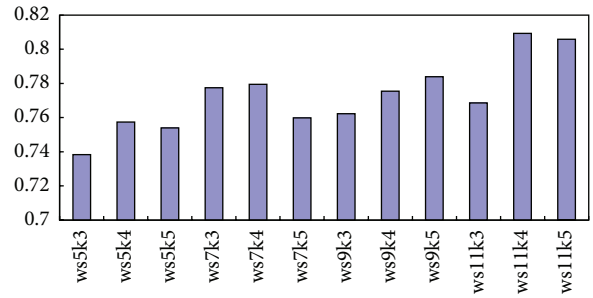

Figure 5: The value of Sp for BalanceCascade learning algorithm in testing dataset.

and Aggre_Balance are the highest compared to others, the Sp is 79.46%, MCC is 0.42 for the Aggre_Easy, the Sp is 70.32%, MCC is 0.42 for the Aggre_Balance. It indicates that our predictor has good performance to predict the positive samples in the imbalance dataset. However, the value of specificity is lower than others. For Aggre_Easy, the value of specificity (Sp = 74.43%) is lower than Amyloidogenic Pattern, Average Packing Density, Beta-strand contiguity, SecStr, Tango, AMYLPRED, and AMYLPRED2, slightly lower than Aggrescan, AmyloidMutants, and Hexapeptide Conf. Energy, and higher than NetCSSP, PaFigure, and Waltz. For Aggre_Balance, the value of specificity (Sp = 80.70%) is lower than Amyloidogenic Pattern, Average Packing Density, Beta-strand contiguity, SecStr, Tango, AMYLPRED, and AMYLPRED2 and higher than other methods. More importantly, the reasonably good performance of Aggre_Easy and Aggre_Balance reflects that the method effectively captures the information of aggregation sites, and we propose that the hybrid classification approach can take advantage of the simplicity of the Bayes technique and the sensitivity of the undersampling ensemble learning algorithm.

In Table 5, the false positives (FP) is large; the main reason was because of the fact that only a relative small portion of them have been studied and confirmed experimentally to be amyloidogenic [11]. On the other hand, we would propose the window redirection operator to improve the prediction performance in the future.

3.3. Web Server for Aggregation-Prone Prediction. An effective prediction servers, Aggre_Easy and Aggre_Balance, are available at http://202.198.129.220:8080/AggrePrediction/. And it is hosted on Apache 2.2 web server by using Windows 2003 server environment. In the web server, the models based on the datasets with the optimal parameters are used to predict sites in submitted sequences. As is displayed in Figures 6 and 7, users could submit the uncharacteristic sequences with FASTA format, and the system would return the prediction results. A region in the polypeptide sequence was considered an aggregation prone if there are 5 or more sequentially continuous residues to be prediction aggregation prone.

4. Conclusion

Accurate identification of the aggregation residues could help fully decipher the molecular mechanisms. Though some researchers have focused on this problem, the overall prediction performance is still not satisfied. In this paper, we develop approaches Aggre_Easy and Aggre_Balance to predict the aggregation prone from the primary sequence information. The Aggre_Easy achieves a promising performance with a sensitivity of 79.47%, a specificity of 80.70%, and a MCC of 0.42; the sensitivity, specificity, and MCC of Aggre_Balance reach 70.32%, 80.70%, and 0.42. Experimental results show that the performances of Aggre_Easy and Aggre_Balance predictor are better than several other state-of-the-art predictors and our methods are helpful for the prediction of aggregation prone.

Table 5: The performance comparison of Aggre_Easy and Aggre_Balance with existing 12 predictors.

Method	Sn (%)	Sp (%)	Q (%)	MCC	TP	TN	FP	FN
Aggrescan	35.37	79.26	57.32	0.13	445	5210	1363	813
AmyloidMutants	41.65	74.91	58.28	0.14	524	4924	1649	734
Amyloidogenic Pattern	13.99	94.95	54.22	0.12	176	6208	365	1082
Average Packing Density	28.70	84.12	56.41	0.12	361	5529	1044	897
Beta-strand contiguity	33.15	85.62	59.39	0.18	417	5628	945	841
Hexapeptide Conf. Energy	39.27	78.69	58.98	0.15	494	5172	1401	764
NetCSSP	51.27	65.22	58.25	0.12	645	4287	2286	613
PaFigure	51.75	71.43	61.59	0.18	651	4695	1878	607
SecStr	11.37	94.40	52.88	0.09	143	6205	368	1115
Tango	13.67	95.57	54.62	0.14	172	6282	291	1086
Waltz	56.44	65.42	60.93	0.16	710	4300	2273	548
AMYLPRED	32.99	86.23	59.61	0.19	415	5668	905	843
AMYLPRED2	39.27	84.48	61.88	0.22	494	5553	1020	764
Aggre_Easy	79.46	74.43	76.95	0.42	1000	4892	1681	258
Aggre_Balance	70.32	80.70	75.51	0.42	885	5304	1269	373

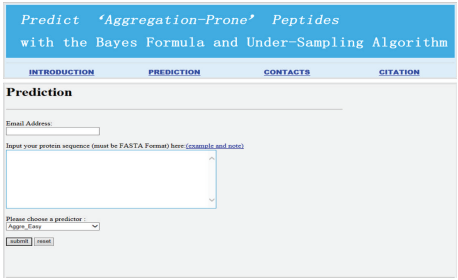

Figure 6: The interface of user input.

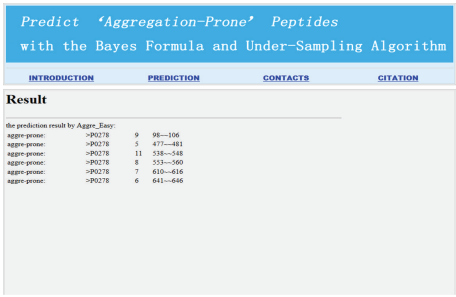

Figure 7: The example of prediction result by Aggre_Easy.

Supporting Information

Text S1: All datasets are consisting of the 33 proteins and their sites information.

Text S2: The prediction result of aggregation prone for Aggre_Easy, Aggre_Balance, AMYLPRED and AMYLPRED2, by comparison. Simply, we remove the single prediction positive site, and, in the future, we will propose the window redirection operator to improve the prediction performance.

Conflict of Interests

The authors declare no conflict of interests.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China (61403077 and 61403076), the Fundamental Research Funds for the Central Universities (14QNJJ029), and the Postdoctoral Science Foundation of China (2014M550166).

References

[1] Z. Zhang, H. Chen, and L. Lai, "Identification of amyloid fibril-forming segments based on structure and residue-based statistical potential," *Bioinformatics*, vol. 23, no. 17, pp. 2218–2225, 2007.

[2] O. Conchillo-Solé, N. S. de Groot, F. X. Avilés, J. Vendrell, X. Daura, and S. Ventura, "AGGRESCAN: a server for the prediction and evaluation of "hot spots" of aggregation in polypeptides," *BMC Bioinformatics*, vol. 8, article 65, 2007.

[3] M. López de la Paz and L. Serrano, "Sequence determinants of amyloid fibril formation," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 1, pp. 87–92, 2004.

[4] O. V. Galzitskaya, S. O. Garbuzynskiy, and M. Y. Lobanov, "Prediction of amyloidogenic and disordered regions in protein chains," *PLoS Computational Biology*, vol. 2, article e177, 2006.

[5] S. Zibaee, O. S. Makin, M. Goedert, and L. C. Serpell, "A simple algorithm locates β-strands in the amyloid fibril core of α-synuclein, Aβ, and tau using the amino acid sequence alone," *Protein Science*, vol. 16, no. 5, pp. 906–918, 2007.

[6] S. J. Hamodrakas, C. Liappa, and V. A. Iconomidou, "Consensus prediction of amyloidogenic determinants in amyloid fibril-forming proteins," *International Journal of Biological Macromolecules*, vol. 41, no. 3, pp. 295–300, 2007.

[7] C. Kim, J. Choi, S. J. Lee, W. J. Welsh, and S. Yoon, "NetCSSP: web application for predicting chameleon sequences and amyloid fibril formation," *Nucleic Acids Research*, vol. 37, no. 2, pp. W469–W473, 2009.

[8] J. Tian, N. Wu, J. Guo, and Y. Fan, "Prediction of amyloid fibril-forming segments based on a support vector machine," *BMC Bioinformatics*, vol. 10, supplement 1, article S45, 2009.

[9] A.-M. Fernandez-Escamilla, F. Rousseau, J. Schymkowitz, and L. Serrano, "Prediction of sequence-dependent and mutational effects on the aggregation of peptides and proteins," *Nature Biotechnology*, vol. 22, no. 10, pp. 1302–1306, 2004.

[10] K. K. Frousios, V. A. Iconomidou, C.-M. Karletidi, and S. J. Hamodrakas, "Amyloidogenic determinants are usually not buried," *BMC Structural Biology*, vol. 9, article 44, 2009.

[11] A. C. Tsolis, N. C. Papandreou, V. A. Iconomidou, and S. J. Hamodrakas, "A consensus method for the prediction of 'Aggregation-Prone' peptides in globular proteins," *PLoS ONE*, vol. 8, no. 1, Article ID e54175, 2013.

[12] Z. Chen, Y.-Z. Chen, X.-F. Wang, C. Wang, R.-X. Yan, and Z. Zhang, "Prediction of ubiquitination sites by using the composition of K-spaced amino acid pairs," *PLoS ONE*, vol. 6, no. 7, Article ID e22930, 2011.

[13] X.-B. Wang, L.-Y. Wu, Y.-C. Wang, and N.-Y. Deng, "Prediction of palmitoylation sites using the composition of k-spaced amino acid pairs," *Protein Engineering, Design and Selection*, vol. 22, no. 11, pp. 707–712, 2009.

[14] Y. Z. Chen, Y. R. Tang, Z. Y. Sheng, and Z. Zhang, "Prediction of mucin-type O-glycosylation sites in mammalian proteins using the composition of k-spaced amino acid pairs," *BMC bioinformatics*, vol. 9, article 101, 2008.

[15] W. Zhang, X. Xu, M. Yin, N. Luo, J. Zhang, and J. Wang, "Prediction of methylation sites using the composition of K-spaced amino acid pairs," *Protein and Peptide Letters*, vol. 20, no. 8, pp. 911–917, 2013.

[16] Y. Xu, J. Ding, L.-Y. Wu, and K.-C. Chou, "iSNO-PseAAC: predict cysteine S-nitrosylation sites in proteins by incorporating position specific amino acid propensity into pseudo amino acid composition," *PLoS ONE*, vol. 8, no. 2, Article ID e55844, 2013.

[17] X. Zhao, W. Zhang, X. Xu, Z. Ma, and M. Yin, "Prediction of protein phosphorylation sites by using the composition of k-spaced amino acid Pairs," *PLoS ONE*, vol. 7, no. 10, Article ID e46302, 2012.

[18] L. H. Lee, R. Rajkumar, and D. Isa, "Automatic folder allocation system using Bayesian-support vector machines hybrid classification approach," *Applied Intelligence*, vol. 36, no. 2, pp. 295–307, 2012.

[19] D. Isa, V. P. Kallimani, and L. H. Lee, "Using the self organizing map for clustering of text documents," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9584–9591, 2009.

[20] D. Isa, L. H. Lee, V. P. Kallimani, and R. Rajkumar, "Text document preprocessing with the bayes formula for classification using the support vector machine," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 9, pp. 1264–1272, 2008.

[21] D. Isa, L. H. Lee, and V. P. Kallimani, "A polychotomizer for case-based reasoning beyond the traditional Bayesian classification approach," *Journal of Computer and Information Science*, vol. 1, no. 1, pp. 57–68, 2008.

[22] X. Liu, J. Wu, and Z. Zhou, "Exploratory under-sampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 2, pp. 539–550, 2008.

[23] T. Liu, "Feature selection based on mutual information for gear faulty diagnosis on imbalanced dataset," *Journal of Computational Information Systems*, vol. 8, no. 18, pp. 7831–7838, 2012.

[24] G. Sang, L. Gao, and Z. Liu, "A Bias-ensemble learning algorithm for imbalanced data processing," *Journal of Computational Information Systems*, vol. 9, no. 5, pp. 2025–2032, 2013.

[25] P. Huang and M. H. Yin, "An upper (lower) bound for Max (Min) CSP," *Science China Information Sciences*, vol. 7, pp. 1–9, 2014.

[26] X. Li and M. Yin, "Multiobjective binary biogeography based optimization for feature selection using gene expression data," *IEEE Transactions on Nanobioscience*, vol. 12, no. 4, pp. 343–353, 2013.

[27] X. Li and M. H. Yin, "Modified differential evolution with self-adaptive parameters method," *Journal of Combinatorial Optimization*, 2014.

[28] X. Li, J. Wang, J. Zhou, and M. Yin, "A perturb biogeography based optimization with mutation for global numerical optimization," *Applied Mathematics and Computation*, vol. 218, no. 2, pp. 598–609, 2011.

[29] X. Li and M. Yin, "A hybrid cuckoo search via Lévy flights for the permutation flow shop scheduling problem," *International Journal of Production Research*, vol. 51, no. 16, pp. 4732–4754, 2013.

[30] J. Zhou, P. Huang, M. Yin, and C. Zhou, "Phase transitions of EXPSPACE-complete problems," *International Journal of Foundations of Computer Science*, vol. 21, no. 6, pp. 1073–1088, 2010.

[31] J. P. Zhou, M. H. Yin, and C. G. Zhou, "New worst upper bound for #SAT," in *Proceedings of AAAI*, 2010.

[32] J. Zhou, M. Yin, X. Li, and J. Wang, "Phase transitions of EXPSPACE-complete problems: a further step," *International Journal of Foundations of Computer Science*, vol. 23, no. 1, pp. 173–184, 2012.

[33] L. Hu, P. Soderhjelm, and U. Ryde, "On the convergence of QM/MM energies," *Journal of Chemical Theory and Computation*, vol. 7, no. 3, pp. 761–777, 2011.

[34] L. Hu, P. Söderhjelm, and U. Ryde, "Accurate reaction energies in proteins obtained by combining QM/MM and large QM calculations," *Journal of Chemical Theory and Computation*, vol. 9, no. 1, pp. 640–649, 2013.

[35] X. Zheng, L. H. Hu, X. J. Wang, and G. H. Chen, "A generalized exchange-correlation functional: the Neural-Networks approach," *Chemical Physics Letters*, vol. 390, no. 1–3, pp. 186–192, 2004.

[36] H. Z. Li, L. Li, Z. Y. Zhong, Y. Han, L. Hu, and Y. H. Lu, "An accurate and efficient method to predict Y-NO bond homolysis bond dissociation energies," *Mathematical Problems in Engineering*, vol. 2013, Article ID 860357, 10 pages, 2013.

[37] H. Z. Li, L. H. Hu, W. Tao et al., "A promising tool to achieve chemical accuracy for density functional theory calculations on Y-NO homolysis bond dissociation energies," *International Journal of Molecular Sciences*, vol. 13, no. 7, pp. 8051–8070, 2012.