Front. Comput. Sci.,2015, 9(4): 643–651
DOI 10.1007/s11704-014-4089-3

RESEARCH ARTICLE

Identification of cytokine via an improved genetic algorithm

Xiangxiang ZENG, Sisi YUAN, Xianxian HUANG, Quan ZOU (✉)

Department of Computer Science, Xiamen University, Xiamen 361005, China

Abstract With the explosive growth in the number of protein sequences generated in the postgenomic age, research into identifying cytokines from proteins and detecting their biochemical mechanisms becomes increasingly important. Unfortunately, the identification of cytokines from proteins is challenging due to a lack of understanding of the structure space provided by the proteins and the fact that only a small number of cytokines exists in massive proteins. In view of fact that a proteins sequence is conceptually similar to a mapping of words to meaning, *n*-gram, a type of probabilistic language model, is explored to extract features for proteins. The second challenge focused on in this work is genetic algorithms, a search heuristic that mimics the process of natural selection, that is utilized to develop a classifier for overcoming the protein imbalance problem to generate precise prediction of cytokines in proteins. Experiments carried on imbalanced proteins data set show that our methods outperform traditional algorithms in terms of the prediction ability.

Keywords *n*-grams, genetic algorithm, cytokine identification, sampling, imbalanced data

1 Introduction

Cytokines are proteins or micromolecular polypeptides, and are also seen as small signaling molecules for cell signaling. With the development of Bioinformatics, research into the identification of cytokines has grown rapidly because it is commonly believed that cytokines can be of great help in preventing, diagnosing, and curing diseases, especially on tumor, inflammation and hematopoietic disorder. Therefore, recognizing cytokines from protein data sets are vital for further studies into cytokines.

Received March 4, 2014; accepted May 22, 2014

E-mail: zouquan@xmu.edu.cn

Cytokines are a category of small proteins and methods for classifying general proteins deliver a great help on the identification of cytokines. Previous research into the classification of proteins has been developed by analyzing sequence structures and then predicting them using self-built or basic classifiers such as support vector machines (SVM) and random forest [1]. These research methods are as follows:

- Hidden markov model (HMM) [2] and artificial neutral network (ANN) [3], both are based on statistical learning theory.
- Basic local alignment search tool (BLAST) [4] and FASTA [5] are sequence alignments based on similarity.
- Emphasizing machine learning techniques, CTKPred [6] and a prediction method proposed by Liu et al. [7], are both based on SVMs. Some other methods use ensemble classifiers [8] or hierarchical multi-label classifiers [9] to get precise prediction.

Additionally, some web-servers has been constructed to provide a powerful and efficient way to operate the prediction [8,10].

In our approach, the analysis of protein sequence structure and machine learning techniques is emphasized to identify cytokines from proteins. We introduce a probabilistic language model called *n*-gram to extract protein features as a preparation for predicting cytokines, following this, we use a self-built classifier that aggregates a sampling model and basic classifiers to classify the imbalanced protein data sets and provide the final predictions.

As a preliminary, we know that a sequential model is typically used for representing proteins, and in the model, the structure and function of proteins are encoded by characteristic amino acid sequences. However, a fundamental unsolved problem is: when we obtain a primary protein sequence, how do we find its 3-dimensional structure and its function in a complex cellular environment? Sequence similarity-based-search tools are usually used to predict the above mentioned problem, e.g., BLAST [4]. However, this method does not work when query proteins that do not have significant sequence similarity to the properties of known proteins. *n*-gram analysis, which has been proven to be exceedingly successful in the domain of natural language, and is directly responsible for progress in automatic speech recognition, text classification, information extraction, statistical machine translation, and other successful challenging tasks in the past four decades. Moreover, the *n*-gram model have recently been also recommended as an efficient way to analyze protein sequencens [11], and some improved *n*-gram models have recently been developed for protein sequence classification [12]. Hence, we hypothesize that the *n*-grams model can be used to explore the sequence landscape of proteins and provide fundamental feature representation to identify cytokines during classification.

However, the number of cytokines is many fewer than the large number of proteins, which makes the problem of cytokine identification an imbalanced binary classfication problem. Although cytokine identification has been widely studied in academia and a variety of methods have been developed but there is still no work that considers the affect of imbalanced data influence on the prediction.

The key to addressing the problem of imbalance is to sample the data set in such a way to form a balanced training set. Previous studies on sampling usually select the desired training set at random, which is easily influenced by noise or ignores the information from unselected data. In our approach, the sampling technique is developed using a genetic algorithm [13]. Using a genetic algorithm, natural selection is imitated across a large number of groups and the best one groups survive and evolve over a certain number of generations. In particular, more improvements are exploited to make this sampling technique more stable and converge quickly. Based on the sampling technique, a self-built classifier is constructed as the ultimate predictor for cytokine identification.

Section 2 outlines the methods we will use for our experiments. Section 3 shows the results obtained, and finally Section 4 states the conclusions of our work and future research directions.

2 Methods

Several procedures have been developed to address the issues of cytokines identification. These procedures include the extraction of features using the *n*-gram model and building a classifier to predict cytokines. More specifically, a sampling model was built based on genetic algorithms to draw a balanced subset from an imbalanced data set, and then a classifier, called an simple genetic algorithm (SGA) classifier was developed by training the subset to give predictions. Additionally, several improvements were made to strengthen the SGA classifier.

2.1 Feature extraction

An *n*-gram model is a type of probabilistic language model for predicting the next item in a sequence; supposing that the *n*th item is only related to the item before it, and the probability of current sequence is the product of the probability of every item in current sequence. Suppose that a sentence w is a sequence of words (items) $w_1, w_2, \dots, w_n$, the probability of sentence w under the *n*-gram model is described in Eq. (1):

$$P(w) = \prod_{i=1}^{n} P(w_i|w_1 w_2 w_3 \cdots w_{i-1}). \tag{1}$$

Along this line of consideration, we suppose that the occurrence probability of the *i*th word is related to the $N-1$ words instead of only the words before it, then Eq. (1) can be described in Eq. (2):

$$P(w) = \prod_{i=1}^{n} P(w_i|w_{i-N+1} w_{i-N+2} w_{i-N+3} \cdots w_{i-1}). \tag{2}$$

In Eq. (2), the most commonly N values are 1, 2, and 3, called 1-gram, 2-gram and 3-gram, respectively.

- 1-gram: works as a words frequency table that describes the occurrence probability of each word.
- 2-gram: equivalent to a transfer matrix. Provides the occurrence probability for each word when given the word before it.
- 3-gram: is a 3-dimensional transfer matrix that provides the occurrence probability for each word w_i when given 2 specific words w_{i-1} and w_{i-2}.

Proteins can be described as an amino acid sequence in a FASTA format file, and just like the mapping of words to meaning, the mapping of these sequences to structure and function of proteins is conceptually similar. This analogy is being studied by a growing number of researchers, and in our approach we combine 1-gram and 2-gram models to extract features of proteins. The standard 20 amino acids are seen as words of a protein sentence. We compute the appearance frequency for different amino acid items as features. In the 1-gram model, there are 20 kinds of item, and there are 20^2 kinds of items in the 2-gram model. In total, for a protein sequence, there are 420 combinations. Furthermore, we calculated the frequency of occurrence for each combination of a protein sequence, and there are 420 features in total. The algorithm is described as follows in detail.

Thus, word n-gram analysis was applied to biological sequences. In this way, a 1×420 numeric array was obtained for each specific protein sequence, that is, for n protein sequences, we obtain an $n \times 420$ array. We consider that a numeric array can represent the protein sequence and contain enough information for classification. We employ the ensemble classifier to distinguish the arrays and find protein differences. The n-gram model extracts more essential protein features compared to other methods [8], this will be discussed in Section 3.

2.2 Basic SGA classifier

A basic SGA classifier framework is developed in two steps. Building a sampling model based on a genetic algorithm in order to tap a superior balanced training set from the imbalanced data set is a preliminary. Afterwards, the SGA classifier takes shape by training a traditional classifier using the balanced set previously obtained.

We confirm that there is a best balanced training set exists in the imbalanced data set, what matters is that it is extremely difficult to discover if random selection is adopted. Nonetheless, genetic algorithms provide a way to find the approximate best after many generations. That is, a balanced training set that has good prediction performance will be obtained.

Genetic algorithms mimic natural selection, in which we randomly initiate a large number of balanced sets, called groups, as the ancestors. Descendants are generated from

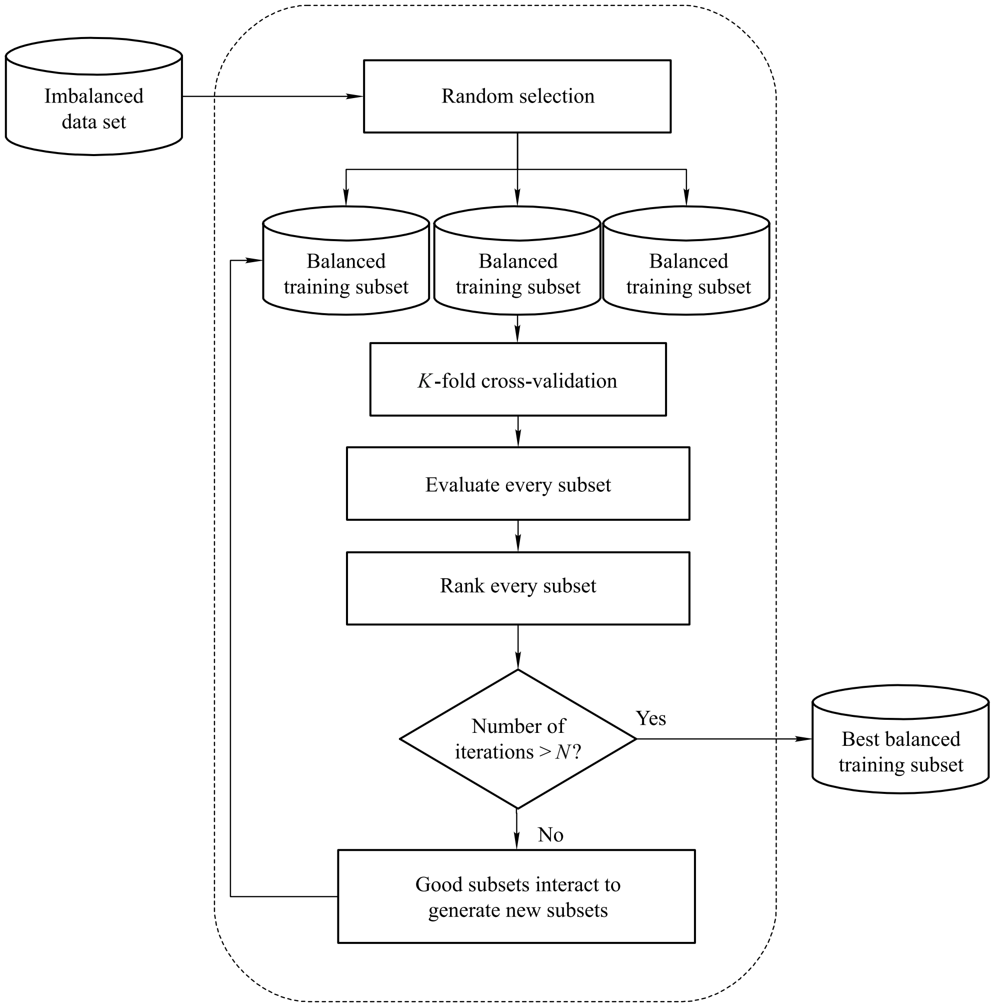


Fig. 1 The sampling model using genetic algorithm

these groups with an interaction rate α, and a fitness function plays a significant part in this procedure to evaluate their prediction ability and discard those that are poor in classification. After many of generations, a best group will be kept and this is the desired balanced set. The procedure is as illustrated in Fig. 1.

In this procedure, a fitness function based on the value classifiers measured is essential to assess every group. Binary classification yields two discrete results: positive and negative. Table 1 shows the four possible outcomes four possible outcomes; if a positive sample is correctly classified , it is counted as a true positive (TP), otherwise false negative (FN), and if negative example is classified correctly, it is counted as a true negative (TN), otherwise a false positive (FP).

Table 1 Confusion matrix

	Predicted positives	Predicted negatives
Real positives	TP	FN
Real negatives	FP	TN

Algorithm 1 Using n-gram to extract features of proteins

```
1:  procedure EXTRACTFEATURES (int $n$)
2:    $sum \leftarrow 0, i \leftarrow 0, values[] \leftarrow 0$
3:    $seq \leftarrow$ a protein sequence
4:    $aminoAcids \leftarrow 20$ amino acid abbreviations
5:    while $i < n$ do
6:      $sum \leftarrow sum + 20^{i+1}$
7:      $allPermutations[] \leftarrow permutation(aminoAcids, i + 1)$
8:    end while
9:    $i \leftarrow 0$
10:   while $i < n$ do
11:     for $(j = 0; j < seq.length() - i; ++j)$ do
12:       $str \leftarrow seq.subString(j, j + i)$
13:       for $(k = 0; k < sum; ++k)$ do
14:         if $str = allPermutations[k]$ then
15:           $values[k] \leftarrow values[i] + 1$
16:         end if
17:       end for
18:     end for
19:   end while
20:   return $values[\,]$
20: end procedure
```

Based on the confusion matrix, the accuracy of the result can be calculated by a fitness function that is the success rate of the prediction result on the test set, defined by Eq. (3).

$$f = \text{Accuracy} = \frac{TP + TF}{TP + TN + FP + FN}. \tag{3}$$

Based on the sampling model, we can generate a balanced training set. This training set is used to build a classifier like, for example a decision tree, and then it can be used to predict the class of test set. The basic SGA classifier is illustrated in Fig. 2.

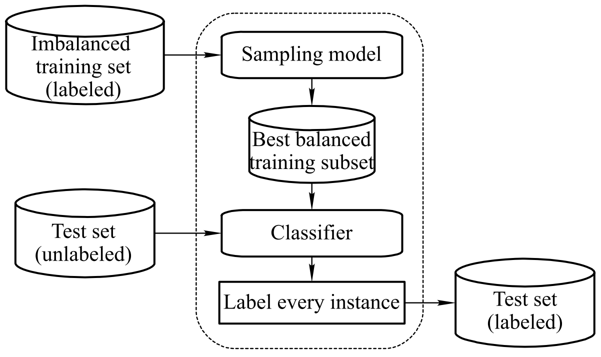


Fig. 2 The basic SGA classifier

2.3 Improved SGA classifier

In order to improve the performance of the SGA classifier, two methods were utilized. First a new fitness function replaces Eq. (4), and a voting classifier with discriminative prediction tables contributes to the classification result.

Equation (3) places great value on the success rate of total prediction of the test set, in imbalanced binary classification problems, however, correctly classifying examples from the minority class (true positives) can be more important than correctly classifying examples from the majority class (true negatives). In other words, all the instances in the test set can be predicted as the majority class will result in a high but useless success rate. Therefore, the true positive (TP) and true negatives (TN) should be considered and a weighted average in Eq. (4).

$$f = \omega * \frac{TP}{TP + FN} + (1 - \omega) * \frac{TN}{TN + FP}. \tag{4}$$

In this equation, the classification accuracy of both classes is treated equally when $\omega = 0.5$, and minority class will contribute more to the fitness function than majority class when $\omega > 0.5$.

This fitness function is designed to investigate two aspects. The first aspect is how to effectively balance the TP and TN rates for a specific instance of a particular classification task. The second aspect is whether classifiers that produce stronger classification accuracy on the minority class (i.e., ω in favour of the TP rate), will have better classification ability when compared to a classifier whose fitness function treats the accuracy of both classes as equally important.

Futhermore, considering that a single subset sampled from the original data set is not convincing enough, three subsets are sampled instead using that model, as shown in Fig. 3. One

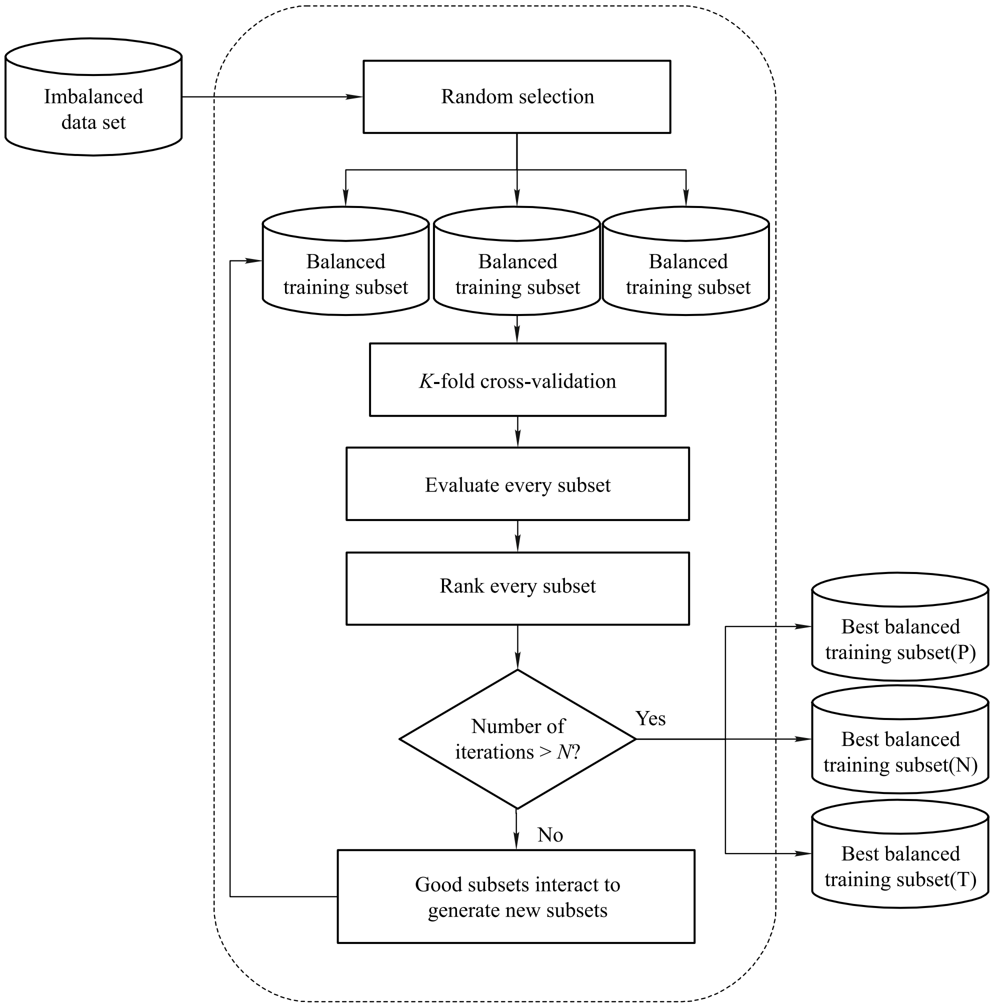


Fig. 3 The improved sampling model using genetic algorithm

of them, named best balanced training subset(T) (BBTS(T)) is inherent in the basic sampling model, and the others, named best balanced training subset(P) (BBTS(P)), whic is precise for positive instances, and best balanced training subset(N) (BBTS(N)), which is precise for negative instances. To distinguish the preoperties of the two subsets, two fitness functions are used to achieve these subsets. Equation (5) concentrates on the number of TPs, all positives for BBTS(P), and in likemanner as Eq. (6), the success rate of prediction on negative instances, to which is attached importance, is specifically for sampling BBTS(N).

$$f_P = \frac{TP}{TP + FP}. \tag{5}$$

$$f_N = \frac{TN}{TN + FN}. \tag{6}$$

Previous improvements on the sampling model aim at achieving three distinctive balanced sets. Using training that separates the internal classifier in SGA will gives its forecast for each instance from the test set. That is, for a particular instance from the test set, there will be three class labels. All possible combinations of the three class labels are shown in Table 2. 0 and 1 are class label the internal classifier in SGA classifier gives, 0 means the prediction result is positive while 1 means negative. P_1 is the prediction result given by training BBTS(T). P_2 is the prediction result given by training BBTS(P). P_3 is the prediction result given by training BBTS(N).

Table 2 Possible combinations of prediction results

No.	P_1	P_2	P_3
1	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

In fact, the final class label is required for every combinations. Conventionally, the final class label of every row should be the same as the one most predicted, as shown in Table 3. Moreover, this rule of getting the final class label can be described by Eq. (7).

$$FP_{\text{basic}} = \lceil \frac{P_1 + P_2 + P_3 - 1.5}{3} \rceil. \tag{7}$$

Table 3 Basic prediction table (FP is the final class label)

No.	P_1	P_2	P_3	FP
1	0	0	0	0
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	1
8	1	1	1	1

Even so, this basic approach to give a final class label does not take full advantage of the properties of BBTS(P) and BBTS(N). Take BBTS(P) for example, its ability to clasify positive instances is nonnegligible, which means it is very influential when it labels "0" to a specific instance. Simultaneously, labeling "1" under training BBTS(N) should be taken with a higher weight. Similarly important, the prediction ability of three distinctive training sets should be assessed to decide the weighting we should apply.

Because BBTS(P) focuses on positive classification and BBTS(N) specializes in negative classification, BBTS(T) is selected as the standard of balancing the classification ability in both cases. In other words, the prediction ability in BBTS(P)'s and BBTS(N)'s expertise should be better than that of BBTS(T)'s. Therefore, three situations should be considered as follows:

1) The prediction ability of BBTS(P) > BBTS(N) $\geqslant$ BBTS(T).
2) The prediction ability of BBTS(N) > BBTS(P) $\geqslant$ BBTS(T).
3) The prediction ability of BBTS(P) $\approx$ BBTS(N) $\geqslant$ BBTS(T).

For this purpose, Eq. (7) is improved to Eq. (8).

$$FP_{\text{improved}} = \lceil \frac{(3 - \omega_1 - \omega_2)P_1 + \omega_1 P_2 + \omega_2 P_3 - 1.5}{3} \rceil. \tag{8}$$

For the three situation, the parameters take the following values,

1) $\omega_1 = 1.6, \omega_2 = 0.9$;
2) $\omega_1 = 0.9, \omega_2 = 1.6$;
3) $\omega_1 = 1, \omega_2 = 1$.

In the situations mentioned above, three logical expressions were explored to give final classification results.

1) $FP_1 = P_1P_2 + P_2P_3$;
2) $FP_2 = P_3 + P_1P_2$;
3) $FP_3 = P_1P_2 + P_1P_3 + P_2P_3$.

Therefore, a new prediction table was developed as seen in Table 4, and the improved SGA classifier framework is shown in Fig. 4.

Table 4 Improved prediction table (FP_1 is the final class label of situation 1); FP_2 is the final class label of situation 2); FP_3 is the final class label of situation 3))

No.	P_1	P_2	P_3	FP_1	FP_2	FP_3
1	0	0	0	0	0	0
2	0	0	1	0	1	0
3	0	1	0	0	0	0
4	0	1	1	1	1	1
5	1	0	0	0	0	0
6	1	0	1	0	1	1
7	1	1	0	1	1	1
8	1	1	1	1	1	1

2.4 Measurement

In the same manner introduced in Section 2.2, the most frequently used metric is accuracy, defined by Eq. (3). However, accuracy is not a useful measure for imbalanced data, especially when there are many fewer instances of the minority class than the majority class. In the simplest case, if a given data set includes 1 positive instance and 99 negatives, a naive approach to classifying examples to be a majority class would be to reach an accuracy of 99 percent. On the surface, the accuracy rate of 99 percent across the entire data set appears superb instead of reflecting the fact that none of minority instances are successfully identified.

Several measures have been developed to take into account the imbalance nature of the problems [14]. Given the confusion matrix, the true positive (TP) rate, also called as recall, and true negative (TN) rate are defined in Eqs. (9) and (10):

$$TP_{\text{rate}} = \frac{TP}{TP + FN}. \tag{9}$$

$$TN_{\text{rate}} = \frac{TN}{TN + FP}. \tag{10}$$

$$FP_{\text{rate}} = \frac{FP}{TN + FP}. \tag{11}$$

$$FN_{\text{rate}} = \frac{FN}{TP + FN}. \tag{12}$$

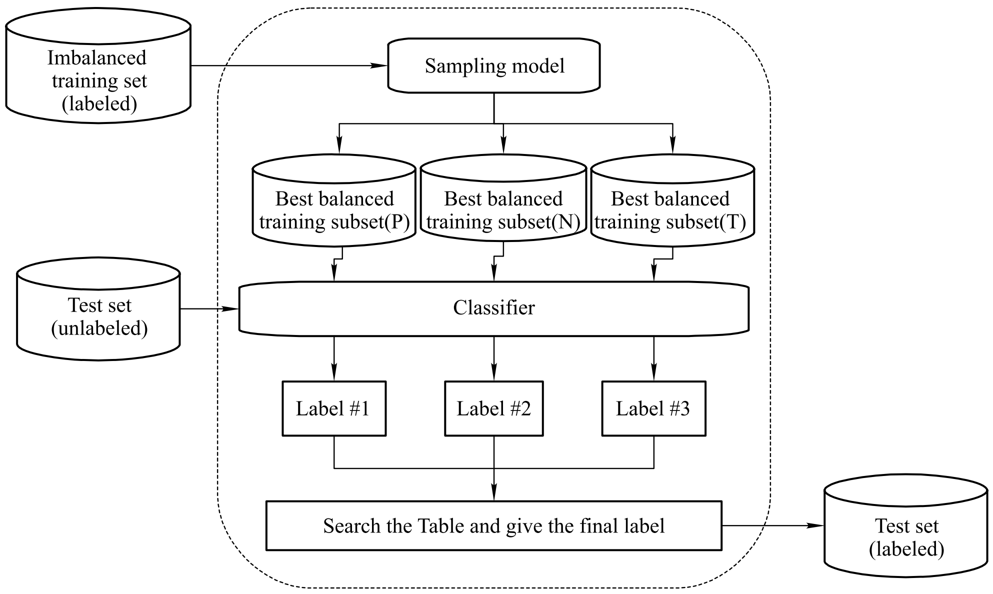

Imbalanced training set (labeled)
Sampling model
Best balanced training subset(P)
Best balanced training subset(N)
Best balanced training subset(T)
Test set (unlabeled)
Classifier
Label #1
Label #2
Label #3
Search the Table and give the final label
Test set (labeled)

Fig. 4 The Improved SGA Classifier

$$\text{Precision} = \frac{TP}{TP+FP}. \tag{13}$$

From these basic metrics, others have been proposed, such as the F-measure [15] or, if we are concerned about the performance of both negative and positive classes the G-mean measure [16]:

$$\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{14}$$

$$\text{G-mean} = \sqrt{TP_{\text{rate}} \times TN_{\text{rate}}}. \tag{15}$$

Furthermore, many classifiers are subject to some kind of threshold that can be varied to achieve different values of the above measures. Each threshold value generates a pair of measurements of $(FP_{\text{rate}}, TP_{\text{rate}})$. By linking these measurements with the FP_{rate} on the X-axis and the TP_{rate} on the Y-axis, an receiver operating characteristic (ROC) [17] graph is plotted. The ideal model is one that obtains 1 TP rate and 0 FP rate. An ROC curve gives a good summary of the performance of a classification model. To compare several classification models by comparing ROC curves, it is hard to claim a winner unless one curve clearly dominates the others over the entire space [18]. The area under an ROC curve (AUC) provides a simple measure of classifier performance for evaluating which model is better on average.

3 Experiments and discussion

3.1 Imbalanced data set

The protein family database (PFAM) is a large collection of protein families and domains [19], in which we simply divide them into two classes, cytokines and non-cytokines. To distinguish cytokines family members from PFAM, seven steps need to be completed as follows:

1) Download cytokines from UniProt (http://www.uniprot.org/);
2) Delete duplicate cytokines;
3) Determine and store the PFAM ID for every cytokine sequence;
4) Delete duplicate PFAM IDs;
5) Choose the longest cytokine sequence as a positive instance for every PFAM ID;
6) Delete all the positive PFAM IDs from PFAM;
7) Choose the longest cytokines sequence as a negative instance for every negative PFAM ID.

In this way, a 126 cytokine sequence was obtained for positive instances and a 10 260 protein sequence for the negative class. With ratio 1 : 81.43, an imbalanced data set for identifying cytokines took shape.

3.2 Selection of parameters and prediction table

In a natural selection process, the inheritable variation of chromosome variation and gene mutation bears the responsibility of delivering gene information to the next generation. Chromosome variation usually emerges in the parents genes for their child through mutation.

In a genetic algorithm, this kind of variation must be introduced. The mutation rate is set to α. We varied α from 0% to

20%. Additionally, as mentioned in Section 2.3, parameter ω in the fitness function and the prediction table in SGA classifier need to be preassigned. To study their impact on the final classification performance, in this group of experiments, we varied ω from 0 to 1, and chose the three individual prediction tables to test which has better performance.

As shown in Table 5, the highest AUC value with least fluctiation is achieved when $\alpha = 0.15$.

Table 5 Classification result using different gene mutation rates α

α	Accuracy	G-mean	AUC
0.00	0.56 ± 0.13	0.63 ± 0.04	0.62 ± 0.02
0.05	0.50 ± 0.12	0.62 ± 0.05	0.64 ± 0.03
0.10	0.56 ± 0.15	0.64 ± 0.07	0.65 ± 0.04
0.15	0.57 ± 0.15	0.64 ± 0.05	0.65 ± 0.03
0.20	0.57 ± 0.15	0.64 ± 0.05	0.65 ± 0.04

As shown in Table 6, the highest AUC value with least fluctuation is achieved when $\omega = 0.5$.

Table 6 Classification result using new fitness function with different weights

ω	Accuracy	G-mean	AUC
0.3	0.57 ± 0.03	0.64 ± 0.04	0.65 ± 0.04
0.4	0.59 ± 0.01	0.65 ± 0.03	0.65 ± 0.04
0.5	0.60 ± 0.05	0.64 ± 0.03	0.65 ± 0.03
0.6	0.60 ± 0.06	0.64 ± 0.03	0.64 ± 0.03
0.7	0.59 ± 0.02	0.65 ± 0.07	0.66 ± 0.08

We performed an experiment using the prediction tables as shown in Table 4, the results are given in Table 7, where we can see that FP_1 achieves a higher AUC value.

Table 7 Classification result using different prediction table (FP_1, FP_2, and FP_3 refer to Table 4, which aggregates the three kinds of prediction table mentioned in Section 2.3)

Prediction Table	Accuracy	G-mean	AUC
FP_1	0.74 ± 0.05	0.68 ± 0.02	0.69 ± 0.02
FP_2	0.82 ± 0.04	0.67 ± 0.01	0.68 ± 0.01
FP_3	0.77 ± 0.07	0.68 ± 0.03	0.68 ± 0.02

3.3 Comparing basic SGA with improved SGA and other software tools

To demonstrate the performance of the improved SGA compared with basic SGA, we conducted experiments to compare their effects. The results are given in Table 8, which illustrates that the improved SGA outperforms basic SGA in all the three aspects (ACC, GM, AUC).

There are only a few software tools or web servers available on line that can predict cytokines from protein primary sequences. We compare our algorithm (CytoSGA) against CTKPred [6] and CytoKey[1]. CTKPred was proposed for identifying cytokine using SVM. It extracts features from depeptide composition and makes comparisons using PFAM search. We compare our CytoPre with CytoKey and CTKPred. Experiments show our system can outperform the other approaches, as shown in Table 9.

Table 8 Classification result using basic and improved SGA (The basic one use the regular fitness funtion; the improved SGA classifier works with $\alpha = 0.15, \omega = 0.5$, and using FP_1 as its prediction table)

SGA	Accuracy	G-mean	AUC
Basic	0.57 ± 0.15	0.64 ± 0.05	0.65 ± 0.03
Improved	0.74 ± 0.05	0.68 ± 0.02	0.69 ± 0.02

Table 9 Classification result comparing with different software tools

Feature extraction method	Accuracy	G-mean	AUC
CTKPred	0.66 ± 0.04	0.64 ± 0.03	0.64 ± 0.03
CytoKey	0.69 ± 0.04	0.65 ± 0.02	0.67 ± 0.02
CytoSGA	0.74 ± 0.05	0.68 ± 0.02	0.69 ± 0.02

4 Conclusion

Our propsed cytokine identification method employs a genetic algorithm to sample from the negative set. This improved genetic algorithm can discover representative negative samples and improve the generalization of the classifier. It solves the imbalanced data classification problem, and increases the cytokine detection performance. However, sampling loses the negative data information. Our method may not be suitable for other applications. Imbalanced classification for big data is still an open problem to be addressed in the future.

The JAVA code of this work is freely available online[2].

Acknowledgements The work was supported by the National Natural Science Foundation of China (Grant Nos. 61371179, 61202011 and 61370010).

References

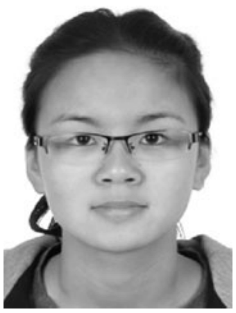
1. Zou Q, Li X, Jiang Y, Zhao Y, Wang G. BinMemPredict: a Web server and software for predicting membrane protein types. Current Proteomics, 2013, 10(1): 2–9
2. Yabuki Y, Muramatsu T, Hirokawa T, Mukai H, Suwa M. GRIFFIN: a system for predicting GPCR-G-protein coupling selectivity using a support vector machine and a hidden Markov model. Nucleic AcidsResearch, 2005, 33(suppl 2): W148–W153
3. Nielsen H, Engelbrecht J, Brunak S, Heijne G V. A neural network method for identification of prokaryotic and eukaryotic signal peptides

[1] http://www.biomed.ecnu.edu.cn/CN/GPCR/Tools/BioAnalysis tools/CytoRAP/CytoKey.aspx
[2] http://datamining.xmu.edu.cn/~xzeng/GA-src.zip

and prediction of their cleavage sites. International Journal of Neural Systems, 1997, 8(5–6): 581–599

4. Altschul S F, Gish W, Miller W, Myers E W, Lipman D J. Basic local alignment search tool. Journal of Molecular Biology, 1990, 215(3): 403–410
5. Pearson W R. Searching protein sequence libraries: comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. Genomics, 1991, 11(3): 635–650
6. Huang N, Chen H, Sun Z. CTKPred: an SVM-based method for the prediction and classification of the cytokine superfamily. Protein Engineering Design and Selection, 2005, 18(8): 365–368
7. Liu B,Wang X, Lin L, Tang B, Dong Q,Wang X. Prediction of protein binding sites in protein structures using hidden Markov support vector machine. BMC bioinformatics, 2009, 10(1): 381
8. Lin C, Zou Y, Qin J, Liu X, Jiang Y, Ke C, Zou Q. Hierarchical classification of protein folds using a novel ensemble classifier. PloS one, 2013, 8(2): e56499
9. Zou Q, Chen W, Huang Y, Liu X, Jiang Y. Identifying multi-functional enzyme by hierarchical multi-label classifier. Journal of Computational and Theoretical Nanoscience, 2013, 10(4): 1038–1043
10. Chou K C, Shen H B. Recent advances in developing web-servers for predicting protein attributes. Natural Science, 2009, 1(2): 63–92
11. Ganapathiraju M, Weisser D, Rosenfeld R, Carbonell J, Reddy R, Klein-Seetharaman J. Comparative *n*-gram analysis of whole-genome protein sequences. In: Proceedings of the 2nd International Conference on Human Language Technology Research. 2002, 76–81
12. Srinivasan S M, Vural S, King B R, Guda C. Mining for class-specific motifs in protein sequence classification. BMC Bioinformatics, 2013, 14(1): 96
13. Koza J R. Genetic Programming. MIT press, 1992
14. Sun Y, Kamel M S, Wong A K, Wang Y. Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, 2007, 40(12): 3358–3378
15. Lewis D, Gale W. Training text classifiers by uncertainty sampling. In: Proceedings of the 14th ACM SIGIR Conference on Research and Development in Information Retrieval. 1994.
16. Kubat M, Holte R C, Matwin S. Machine learning for the detection of oil spills in satellite radar images. Machine learning, 1998, 30(2–3): 195–215
17. Fawcett T. An introduction to ROC analysis. Pattern recognition letters, 2006, 27(8): 861–874
18. Provost F J, Fawcett T. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining. 1997, 97: 43–48
19. Bateman A, Coin L, Durbin R, Finn R D, Hollich V, Griffiths-Jones S, Khanna A, Marshall M, Moxon S, Sonnhammer E L L, Studholme D J, Yeats C, Eddy, S. R. The Pfam protein families database. Nucleic Acids Research, 2004, 32: D138–D141

Xiangxiang Zeng received his BS degree in automation from Hunan University, China in 2005, and his PhD in systems engineering from Huazhong University of Science and Technology, China in 2011. From 2010 to 2011 he spent one year working in the group of natural computing in Seville University, Spain. Currently, he is an assistant professor in the Department of Computer Science, Xiamen University, China. His main research interests include membrane computing, neural computing and automaton theory.



Sisi Yuan is a Master student of the Department of Computer Science at Xiamen University, China. She received her BS degree in software engineering from Hangzhou Dianzi University, China. Her research interests include data mining and bioinformatics.

Xianxian Huang is an undergraduate student of the Department of Computer Science at Xiamen University, China. His main research interests are data mining and bioinformatics.

Quan Zou is an associate professor of computer science at Xiamen University, China. He received his PhD degree from Harbin Institute of Technology, China in 2009. His research is in the areas of bioinformatics, machine learning and parallel computing. Now his focus is on genome assembly, annotation, and functional analysis from next generation sequencing data with parallel computing methods. Several related works have been published in Briefings in Bioinformatics, Bioinformatics, PLOS ONE, and IEEE/ACMTransactions on Computational Biology and Bioinformatics. He serves on many impactful journals and the National Natural Science Foundation of China.