Available online at www.sciencedirect.com

SCIENCE DIRECT®

Biochemical and Biophysical Research Communications 334 (2005) 213–217

BBRC

www.elsevier.com/locate/ybbrc

Boosting classifier for predicting protein domain structural class

Kai-Yan Feng [a], Yu-Dong Cai [b], Kuo-Chen Chou [c,*]

[a] *Imaging Science and Biomedical Engineering, Medical School, The University of Manchester, Manchester, M13 9PT, UK*
[b] *Biomolecular Sciences Department, University of Manchester Institute of Science and Technology, Post Box 88, Manchester, M60 1QD, UK*
[c] *Gordon Life Science Institute, 13784 Torrey Del Mar, San Diego, CA 92130, USA*

Received 9 June 2005
Available online 27 June 2005

Abstract

A novel classifier, the so-called "LogitBoost" classifier, was introduced to predict the structural class of a protein domain according to its amino acid sequence. LogitBoost is featured by introducing a log-likelihood loss function to reduce the sensitivity to noise and outliers, as well as by performing classification via combining many weak classifiers together to build up a very strong and robust classifier. It was demonstrated thru jackknife cross-validation tests that LogitBoost outperformed other classifiers including "support vector machine," a very powerful classifier widely used in biological literatures. It is anticipated that LogitBoost can also become a useful vehicle in classifying other attributes of proteins according to their sequences, such as subcellular localization and enzyme family class, among many others.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Domain structural classification; Binary LogitBoost; One-vs-others LogitBoost; AdaBoost; Support vector machines; Neural network

Although the details of the three-dimensional structures of proteins and domains therein are extremely complicated and irregular, their overall folding patterns are surprisingly simple, regular, and strikingly beautiful from the aesthetical point of view [1–4]. Many protein domains often have similar or identical folding patterns even if they are quite different according to their sequences [5–8]. Actually, about three decades ago Levitt and Chothia tried to classify proteins into the following four structural classes: (1) all-α (Fig. 1A) that is formed essentially by α-helices, (2) all-β (Fig. 1B) essentially by β-strands, (3) α/β (Fig. 1C) containing both α-helices and β-strands that are largely interspersed in forming mainly parallel β-sheets, and (4) $\alpha + \beta$ (Fig. 1D) containing also both of the two secondary structure elements that, however, are largely segregated in forming mainly antiparallel β-sheets. The structural class has ever since become an important attribute for characterizing the overall folding type of a protein or its domain.

Prediction of protein structural class is an important topic in protein science (see, e.g., a review [9]). A series of previous studies have shown that some correlation between the protein structural class and amino acid composition does exist. Actually many efforts were made to predict the structural classes of proteins based on their amino acid composition [10–20]. Here we would like to introduce a novel approach, the so-called "LogitBoost" [21], for predicting the protein structural classes. Because an individual domain is the most basic unit in structural classification [22], the present study will focus on protein domains.

Boosting algorithms

Boosting was originally proposed to combine several weak classifiers together to improve the classification performance. Boosting has been used to solve various

[*] Corresponding author.
E-mail address: kchou@san.rr.com (K.-C. Chou).

0006-291X/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.bbrc.2005.06.075

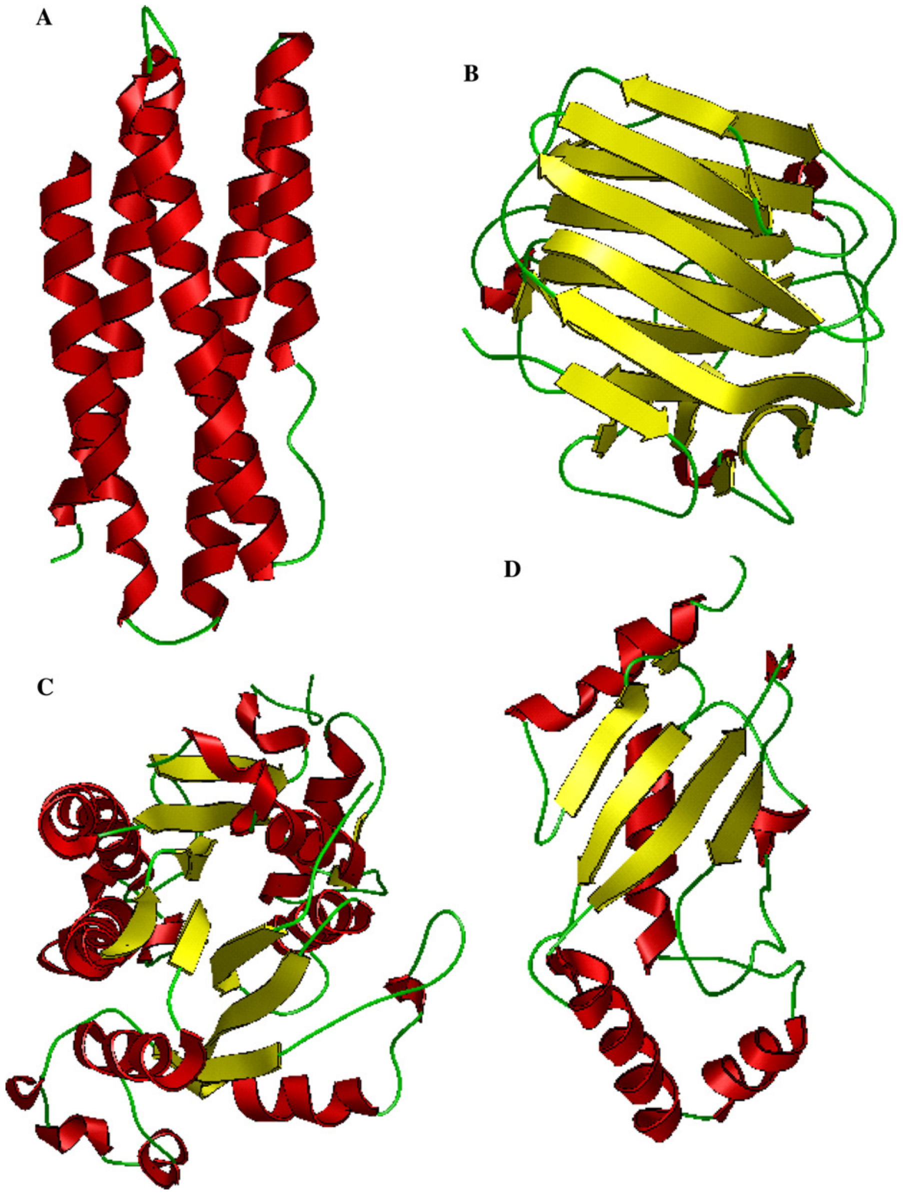

Fig. 1. Ribbon drawings to show the four structural classes of proteins: (A) all-α, (B) all-β, (C) α/β, and (D) α + β. Reproduced from [9] with permission.

classification problems, such as text classification [23], natural language processing [24], and cancer classification [25], among others.

AdaBoost

Of the boosting algorithms, AdaBoost (Adaptive Boosting) is a more capable and practical boosting algorithm that was proposed by Freund and Schapire [26]. AdaBoost is a meta learning algorithm, which tries to build a weak classifier iteratively on others according to the performance of the previous weak classifiers. Accordingly, AdaBoost is driven to focus on the hard samples by putting more weight on them that could otherwise not be correctly classified with the previous weak classifiers. AdaBoost is able to reduce training errors exponentially fast as long as the weak classifiers perform just better than random [26]. It was found [27,28] that AdaBoost had very good generalization (the ability to classify new data). However, like most other classifiers, AdaBoost also had the shortcoming of over-fit problem when dealing with very noisy data [29]. This is because of that, during the process of its operation, AdaBoost can be considered as fitting an additive logistic regression model $F(\vec{x}) = \sum_{t=1}^{T} \alpha_t f_t(\vec{x})$ to minimize the expectation of an exponential loss function $\mathrm{ELOSS}(F) = E(\mathrm{e}^{-yF(\vec{x})})$ [21], and that the exponential loss function changes exponentially with the classification error, rendering the AdaBoost algorithm vulnerable while handling noisy data. To overcome such a problem, Friedman et al. [21] proposed to use LogitBoost that can reduce training errors linearly and hence yield better generalization, as illustrated below.

Binary LogitBoost

In LogitBoost, the following binomial log-likelihood loss function is introduced:

$$\mathrm{LLOSS}(F) = E[-\log(1 + \mathrm{e}^{yF(\vec{x})})] \tag{1}$$

which changes linearly with the classification error and turns out to be less sensitive to noise and outliers. The optimization can be achieved by using Newton steps to fit an additive symmetric logistic model.

The Newton steps of optimizing the loss function can be carried out as an iterative computation, which builds up a robust classifier by iteratively adding another weak classifier. Suppose the input data set is denoted as

$$S = \{(\vec{x}_1, y_1), \dots, (\vec{x}_N, y_N)\}, \quad \vec{x}_i \in X, \quad y_i \in Y = \{-1, 1\}. \tag{2}$$

In each of the iteration steps $t = 1, \dots, T$, the weight w_i and working respond z_i are given by

$$\begin{cases} w_i = p(\vec{x}_i)[1 - p(\vec{x}_i)] \\ z_i = \dfrac{(y_i + 1)/2 - p(\vec{x}_i)}{w_i} \quad (i = 1, 2, \dots, N) \end{cases} \tag{3}$$

with the initial values given by: $w_i = 1/N (i = 1, \dots, N)$, $p(\vec{x}) = P(y = 1|x) = 1/2$, and $F(\vec{x}) = 0$ (cf. Eq. (1)).

Then fit the function $f_t(\vec{x})$ by a weighted least-squares regression of z_i to $\vec{x}_i$ using weights w_i, followed by updating the committee function and the probability according to the following equations:

$$\begin{cases} F(\vec{x}) = F(\vec{x}) + \frac{1}{2} f_t(\vec{x}), \\ p(\vec{x}) = \dfrac{\mathrm{e}^{F(\vec{x})}}{\mathrm{e}^{F(\vec{x})} + \mathrm{e}^{-F(\vec{x})}}. \end{cases} \tag{4}$$

When all the iterations are finished, we have (cf. Eq. (4))

$$F(\vec{x}) = \tfrac{1}{2}[0 + f_1(\vec{x}) + f_2(\vec{x}) + \cdots + f_T(\vec{x})]. \tag{5}$$

Thus, the overall classifier is given by the following decision function:

$$\Delta(\vec{x}) = \mathrm{sign}[F(\vec{x})] = \begin{cases} 1 & \text{if } F(\vec{x}) > 0, \\ -1 & \text{if } F(\vec{x}) < 0, \end{cases} \tag{6}$$

where $\Delta(\vec{x}) = 1$ means $\vec{x}$ belongs to class 1 and $\Delta(\vec{x}) = -1$ means $\vec{x}$ belongs to class 2.

The construction of weak classifiers is one of the key factors affecting the performance of the boosting algorithms. The weak classifier $f_t(\vec{x})$ should be able to cope with reweighing of the data and resistant to over-fit. In our study we use regression decision tree and w_i to fit the data $\{(\vec{x}_1, z_1), \dots, (\vec{x}_N, z_N)\}$. Decision trees try to divide the input space into nested regions, usually rectangles, to minimize the least squares error, which has been practically proved to be one of the most suitable weak classifiers for boosting.

The above LogitBoost is a binary classifier, which can only separate two classes. To deal with a problem with more than two classes, we have to extend the binary LogitBoost as given below.

One-vs-others LogitBoost

Two strategies commonly used to solve the multi-class problem are: (1) the one-vs-others LogitBoost, and (2) the multi-class LogitBoost. However, it is very difficult for the multi-class LogitBoost to define the hard samples. Here we adopt the one-vs-others strategy [30,31], which works quite straightforward as follows. The entire training data set is divided into two sets in turn for each class, with one set of data belonging to the singled-out class and the rest of the data (from all the other classes) belonging to another data set. Thus, if there are k classes, k binary LogitBoost classifiers are built. Since each LogitBoost generates the probability of a testing datum belonging to the class, k binary LogitBoost classifiers will output a vector of classification probability $P(\vec{x}) = [p_1(\vec{x}), p_2(\vec{x}), \dots, p_k(\vec{x})]$. The testing data will be predicted to belonging to the class with the highest probability; i.e., $C(x) = \mathrm{argMax}_i[p_i(x)]$. For the current case, the pair-wise classes are α-vs-others, β-vs-others, (α/β)-vs-others and (α + β)-vs-others.

Implementation

The program for the one-vs-others LogitBoost was downloaded from [25]. However, instead of using stumps, the classification trees with depth three was used that turned out to be much better than stumps because they were able to generate several unconnected regions for a category.

Results and discussion

Two working datasets taken from Zhou [18] were used to demonstrate the power of LogitBoost classifier. The first dataset contains 277 protein domains, of which 70 are all-α domains, 61 all-β, 81 α/β, and 65 α + β. The second dataset contains 498 domains, of which 107 are all-α domains, 126 all-β, 136 α/β, and 129 α + β. The amino acid composition was used to represent the sample of a protein domain [11]. Therefore, each input of the LogitBoost actually corresponds to a vector in a 20-dimensional space [12,14].

The prediction of the LogitBoost classifier was examined by the jackknife test, which is deemed the most rigorous and objective cross-validation test in statistical

Table 1
Jackknife cross-validation success rates by neural network, SVM, and LogitBoost

Dataset	Algorithm	Rate of correct prediction for each class (%)				Overall success rate (%)
		All-α	All-β	α/β	α + β	
277 domains[a]	Neural network	68.6	85.2	86.4	56.9	74.7
	SVM	74.3	82.0	87.7	72.3	79.4
	LogitBoost	81.4	88.5	92.6	72.3	84.1
498 domains[b]	Neural network	86.0	96.0	88.2	86.0	89.2
	SVM	88.8	95.2	96.3	91.5	93.2
	LogitBoost	92.5	96.0	97.1	93.0	94.8

[a] Taken from Table 1 of Zhou [18].
[b] Taken from Table 2 of Zhou [18].

prediction [15]. The success rates thus obtained are given in Table 1, where, for facilitating comparison, the corresponding rates obtained by neural networks [32] and support vector machines (SVM) [33,34] are also listed. As we can see from the table, the current LogitBoost is superior to both the neural network and SVM in identifying the structural classification for the dataset of the 277 protein domains as well as the dataset of 498 domains.

Conclusion

The LogitBoost is a promising classifier as reflected by the fact that its success rates in predicting the protein domain structural classes for the two datasets constructed by previous investigators are even higher than those by the very powerful neural network and SVM approaches. Moreover, it has not escaped our notice that the LogitBoost classifier can also be used to predict other protein attributes, such as subcellular localization [35–40], membrane types [41–45], enzyme family and subfamily classes [46–49], enzyme active sites [50,51], G-protein coupled receptor classification [52,53], and protein quaternary structure types [54], among many others.

References

[1] A.V. Finkelstein, O.B. Ptitsyn, Prog. Biophys. Mol. Biol. 50 (1987) 171–190.
[2] K.C. Chou, L. Carlacci, Proteins: Struct. Funct. Genet. 9 (1991) 280–295.
[3] K.C. Chou, Biochem. Biophys. Res. Commun. 316 (2004) 636–642.
[4] K.C. Chou, Biochem. Biophys. Res. Commun. 319 (2004) 433–438.
[5] J.S. Richardson, Nature 268 (1977) 495–500.
[6] J.S. Richardson, Adv. Protein Chem. 34 (1981) 167–339.
[7] O.B. Ptitsyn, A.V. Finkelstein, Q. Rev. Biophys. 13 (1980) 339–386.
[8] K.C. Chou, Curr. Med. Chem. 11 (2004) 2105–2134.
[9] K.C. Chou, Curr. Protein Pept. Sci. 1 (2000) 171–208.
[10] P. Klein, C. Delisi, Biopolymers 25 (1986) 1659–1672.
[11] J.J. Chou, C.T. Zhang, J. Theor. Biol. 161 (1993) 251–262.
[12] K.C. Chou, C.T. Zhang, J. Biol. Chem. 269 (1994) 22014–22020.
[13] B. Mao, K.C. Chou, C.T. Zhang, Protein Eng. 7 (1994) 319–330.
[14] K.C. Chou, Proteins: Struct. Funct. Genet. 21 (1995) 319–344.
[15] K.C. Chou, C.T. Zhang, Crit. Rev. Biochem. Mol. Biol. 30 (1995) 275–349.
[16] I. Bahar, A.R. Atilgan, R.L. Jernigan, B. Erman, Proteins: Struct. Funct. Genet. 29 (1997) 172–185.
[17] G.P. Zhou, N. Assa-Munt, Proteins: Struct. Funct. Genet. 44 (2001) 57–59.
[18] G.P. Zhou, J. Protein Chem. 17 (1998) 729–738.
[19] Y.D. Cai, Y.X. Li, K.C. Chou, Biochim. Biophys. Acta 1476 (2000) 1–2.
[20] Y.D. Cai, G.P. Zhou, Biochimie 82 (2000) 783–785.
[21] J. Friedman, T. Hastie, R. Tibshirani, Ann. Stat. 337-407. (2000) 337–407.
[22] A.G. Murzin, S.E. Brenner, T. Hubbard, C. Chothia, J. Mol. Biol. 247 (1995) 536–540.
[23] R.E. Schapire, Y. Singer, Mach. Learn. 37 (1999) 297–336.
[24] M. Haruno, S. Shirai, Y. Ooyama, Mach. Learn. 34 (1999) 131–149.
[25] M. Dettling, P. Buhlmann, Bionformatics 19 (2003) 1061–1069.
[26] Y. Freund, R. Schapire, J. Comput. Syst. Sci. 55 (1997) 119–139.
[27] L. Breiman, Ann. Stat. 26 (1998) 801–849.
[28] H. Drucker, C. Cortes, Adv. Neural Inf. Process. Syst. 8 (1996) 479–485.
[29] G. Ratsch, T. Onoda, K.R. Muller, Mach. Learn. 42 (2001) 287–320.
[30] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C. Sugnet, J.M. Ares, D. Haussler, Proc. Natl. Acad. Sci. USA 97 (2000) 262–267.
[31] C.H. Ding, I. Dubchak, Bioinformatics 17 (2001) 349–358.
[32] C. Bishop, Neural Networks for Pattern Recognition, Oxford Press, Oxford, 1995.
[33] N. Cristianini, J. Shawe-Taylor, Support Vector Machines, Cambridge University Press, Cambridge, 2000.
[34] V. Vapnik, Statistical Learning Theory, Wiley-Interscience, New York, 1998.
[35] K.C. Chou, D.W. Elrod, Protein Eng. 12 (1999) 107–118.
[36] G.P. Zhou, K. Doctor, Proteins: Struct. Funct. Genet. 50 (2003) 44–48.
[37] Y.X. Pan, Z.Z. Zhang, Z.M. Guo, G.Y. Feng, Z.D. Huang, L. He, J. Protein Chem. 22 (2003) 395–402.
[38] K.C. Chou, Y.D. Cai, J. Cell. Biochem. 90 (2003) 1250–1260 (Addendum, ibid. 2004, 1291, No.1255, P.1085).
[39] X. Xiao, S. Shao, Y. Ding, Z. Huang, Y. Huang, K.C. Chou, Amino Acids 28 (2005) 57–61.
[40] K.C. Chou, Y.D. Cai, Bioinformatics 21 (2005) 944–950.
[41] K.C. Chou, D.W. Elrod, Proteins: Struct. Funct. Genet. 34 (1999) 137–153.

[42] Y.D. Cai, G.P. Zhou, K.C. Chou, Biophys. J. 84 (2003) 3257–3263.
[43] M. Wang, J. Yang, G.P. Liu, Z.J. Xu, K.C. Chou, Protein Eng. Des. Sel. 17 (2004) 509–516.
[44] M. Wang, J. Yang, Z.J. Xu, K.C. Chou, J. Theor. Biol. 232 (2005) 7–15.
[45] K.C. Chou, Y.D. Cai, J. Chem. Inf. Model. 45 (2005) 407–413.
[46] K.C. Chou, Y.D. Cai, Protein Sci. 13 (2004) 2857–2863.
[47] K.C. Chou, Bioinformatics 21 (2005) 10–19.
[48] K.C. Chou, D.W. Elrod, J. Proteome Res. 2 (2003) 183–190.
[49] K.C. Chou, Y.D. Cai, Biochem. Biophys. Res. Commun. 325 (2004) 506–509.
[50] K.C. Chou, Y.D. Cai, Proteins: Struct. Funct. Genet. 55 (2004) 77–82.
[51] Y.D. Cai, G.P. Zhou, C.H. Jen, S.L. Lin, K.C. Chou, J. Theor. Biol. 228 (2004) 551–557.
[52] D.W. Elrod, K.C. Chou, Protein Eng. 15 (2002) 713–715.
[53] K.C. Chou, D.W. Elrod, J. Proteome Res. 1 (2002) 429–433.
[54] K.C. Chou, Y.D. Cai, Proteins: Struct. Funct. Genet. 53 (2003) 282–289.