Protein Classification with Multiple Algorithms

Sotiris Diplaris[1], Grigorios Tsoumakas[2], Pericles A. Mitkas[1], and Ioannis Vlahavas[2]

[1] Dept. of Electrical and Computer Engineering, Aristotle University of Thessaloniki,
54126, Thessaloniki, Greece,
{diplaris,mitkas}@danae.ee.auth.gr

[2] Dept. of Informatics, Aristotle University of Thessaloniki,
54126, Thessaloniki, Greece,
{greg,vlahavas}@csd.auth.gr

Abstract. Nowadays, the number of protein sequences being stored in central protein databases from labs all over the world is constantly increasing. From these proteins only a fraction has been experimentally analyzed in order to detect their structure and hence their function in the corresponding organism. The reason is that experimental determination of structure is labor-intensive and quite time-consuming. Therefore there is the need for automated tools that can classify new proteins to structural families. This paper presents a comparative evaluation of several algorithms that learn such classification models from data concerning patterns of proteins with known structure. In addition, several approaches that combine multiple learning algorithms to increase the accuracy of predictions are evaluated. The results of the experiments provide insights that can help biologists and computer scientists design high-performance protein classification systems of high quality.

1 Introduction

A crucial issue in bioinformatics is structural biology, i.e. the representation of the structure of several biological macromolecules. The knowledge of the 3D structure of proteins is a strong weapon in combating many diseases, since most of them are caused by malfunctions of the proteins involved in several functions of the human cells. Until now the biological effect of proteins could be identified only by expensive in vitro experiments. In the recent years, though, large databases were created for the recording and exploitation of biological data, due to the human DNA and protein decoding. With the contribution of modern data analysis techniques, such as machine learning and knowledge discovery, the issue has been approached computationally, thus providing fast and more flexible solutions.

The function of a protein is directly related to its structure. Proteins are grouped into several families according to the functions they perform. All proteins contained in a family feature a certain structural relation, thus having similar properties. Patterns are short amino acid chains that have a specific order, while profiles are computational representations of multiple sequence alignments using hidden Markov models.

We will refer to both profiles and patterns as motifs. Motifs have been widely used for the prediction of a protein's properties, since the latter are mainly defined by their motifs. Prosite [1], Pfam [2] and Prints [3] are the most common databases where motifs are being recorded.

Machine learning (ML) algorithms [4] can offer the most cost effective approach to automated discovery of a priori unknown predictive relationships from large data sets in computational biology [5]. A plethora of algorithms to address this problem have been proposed, by both the artificial intelligence and the pattern recognition communities. Some of the algorithms create decision trees [6,7], others exploit artificial neural networks [8] or statistical models [9].

An important issue however that remains is which from the multitude of machine learning algorithms to use for training a classifier in order to achieve the best results. The plot thickens if we also consider the recent advances in ensemble methods that combine several different classification algorithms for increasing the accuracy. This creates a problem to the ML expert who wants to provide the biologist with a good model for protein classification.

In this paper we perform an empirical comparison of the performance of several different classification algorithms for the problem of motif-based protein classification. Moreover, we exploit the combination of different classification algorithms in order to achieve accuracy improvement. Two main paradigms in combining different classification algorithms are used: classifier selection and classifier fusion. The first one selects a single algorithm for classifying a new instance, while the latter combines the decisions of all algorithms.

The rest of this paper is organized as follows. Section 2 presents the biological problem of motif-based protein classification, as well as methods for combining multiple classification algorithms in order to optimize the predictive performance. Section 3 describes the details of the performed classification experiments for the comparison of several different classification algorithms. In Section 4 the results are presented and discussed, and finally, Section 5 concludes this work and points at the future outlooks.

2 Problem Description

2.1 The Motif-Based Protein Classification Problem

The basic problem we are trying to solve can be stated as follows: "Given a set of proteins with known properties (that have been experimentally specified), we aim to induce classifiers that associate motifs to protein families, referred to as protein classes." In Fig.1 this approach is designated.

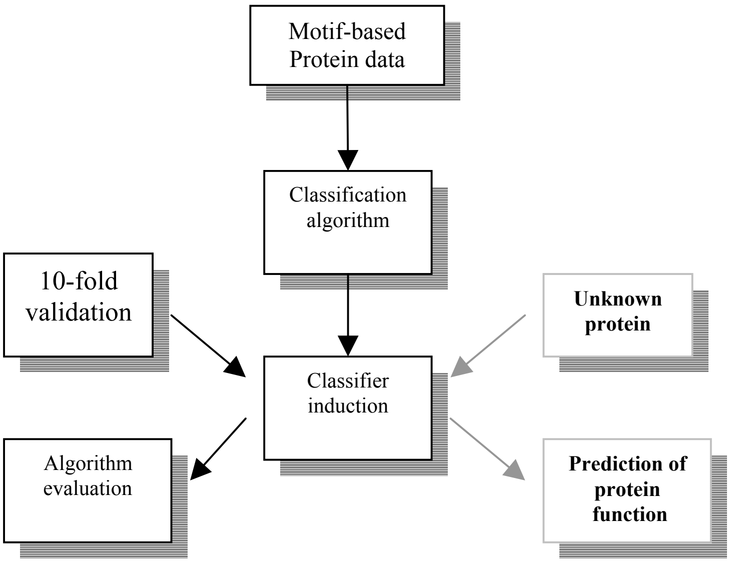

Fig.1. The protein classification problem

Any protein chain can be mapped into a representation based on attributes. Such a representation supports the efficient function of data-driven algorithms, which represent instances as classified part of a fixed set of attributes. A very important issue in the data mining process is the efficient choice of attributes. In our case, protein chains are represented using a proper motif sequence vocabulary [10].

Suppose the vocabulary contains *N* motifs. Any given protein sequence typically contains a few of these motifs. We encode each sequence as an *N*-bit binary pattern where the i^{th} bit is 1 if the corresponding motif is present in the sequence; otherwise the corresponding bit is 0. Each *N*-bit sequence is associated with a label that identifies the functional family of the sequence (if known). A training set is simply a collection of *N*-bit binary patterns each of which has associated with it, a label that identifies the function of the corresponding protein. This training set can be used to train a classifier which can then be used to assign novel sequences to one of the several functional protein families represented in the training set. The data representation procedure is depicted in Figure 2.

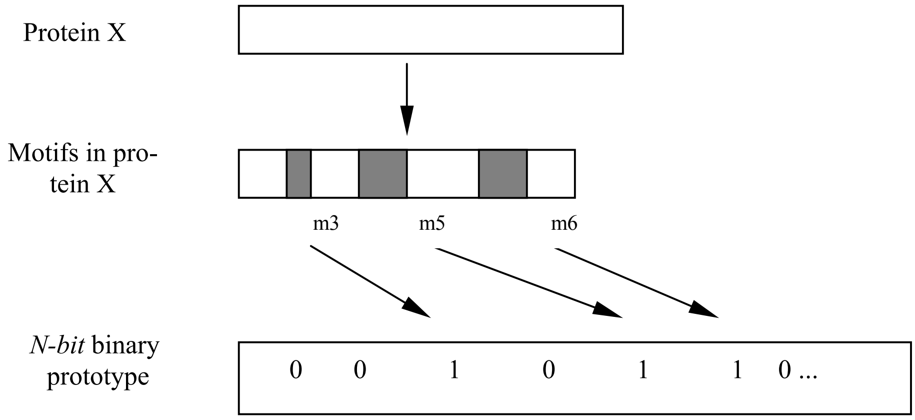

Fig. 2. Data representation.

2.2 Combining Multiple Classification Algorithms

The main motivation for combining different classification algorithms is accuracy improvement. Different algorithms use different biases for generalizing from examples and different representations of the knowledge. Therefore, they tend to err on different parts of the instance space. The combined use of different algorithms could lead to the correction of the individual uncorrelated errors. There are two main paradigms for handling an ensemble of different classification algorithms: *Classifier Selection* and *Classifier Fusion*. The first one selects a *single* algorithm for classifying a new instance, while the latter fuses the decisions of *all* algorithms. This section presents the most important methods from both categories.

Classifier Selection
A very simple method of this category is found in the literature as Evaluation and Selection or SelectBest. This method evaluates each of the classification algorithms (typically using 10-fold cross-validation) on the training set and selects the best one for application on the test set. Although this method is simple, it has been found to be highly effective and comparable to other more complex state-of-the-art methods [11].

Another line of research proposes the selection of a learning algorithm based on its performance on similar learning domains. Several approaches have been proposed for the characterization of learning domain, including general, statistical and information-theoretic measures [12], landmarking [13], histograms [14] and model-based data characterizations [15]. Apart from the characterization of each domain, the performance of each learning algorithm on that domain is recorded. When a new domain arrives, the performance of the algorithms in the k-nearest neighbors of that domain is retrieved and the algorithms are ranked according to their average performance. In [12], algorithms are ranked based on a measure called Adjusted Ratio of Ratios (ARR), that combines accuracy and learning time of algorithm, while in [16], algorithms are ranked based on Data Envelopment Analysis, a multicriteria evaluation technique that can combine various performance metrics, like accuracy, storage space, and learning time.

In [17,18] the accuracy of the algorithms is estimated locally on a number of examples that surround each test example. Such approaches belong to the family of Dynamic Classifier Selection [19] and use a different algorithm in different parts of the instance space.

Two similar, but more complicated approaches that were developed by Merz[20] are Dynamic Selection and Dynamic Weighting. The selection of algorithms is based on their local performance, but not around the test instance itself, rather around the meta-instance comprising the predictions of the classification models on the test instance. Training meta-instances are produced by recording the predictions of each algorithm, using the full training data both for training and for testing. Performance data are produced by running *m* *k*-fold cross-validations, and averaging the *m* evaluations for each training instance.

Classifier Fusion

Unweighted and Weighted Voting are two of the simplest methods for combining not only Heterogeneous but also Homogeneous models. In Voting, each model outputs a class value (or ranking, or probability distribution) and the class with the most votes (or the highest average ranking, or average probability) is the one proposed by the ensemble. Note that this type of Voting is in fact called Plurality Voting, in contrast to the frequently used term Majority Voting, as the latter implies that at least 50% (the majority) of the votes should belong to the winning class. In Weighted Voting, the classification models are not treated equally. Each model is associated with a coefficient (weight), usually proportional to its classification accuracy.

Stacked Generalization [21], also known as Stacking, is a method that combines multiple classifiers by learning a meta-level (or level-1) model that predicts the correct class based on the decisions of the base-level (or level-0) classifiers. This model is induced on a set of meta-level training data that are typically produced by applying a procedure similar to k-fold cross-validation on the training data:

Let D be the level-0 training data set. D is randomly split into k disjoint parts D_1 … D_k of equal size. For each fold i=1..i of the process, the base-level classifiers are trained on the set $D \setminus D_{\mathrm{i}}$ and then applied to the test set $\mathrm{D_i}$. The output of the classifiers for a test instance along with the true class of that instance form a meta-instance.

A meta-classifier is then trained on the meta-instances and the base-level classifiers are trained on all training data *D*. When a new instance appears for classification, the output of all base-level classifiers is first calculated and then propagated to the meta-level classifier, which outputs the final result.

Ting and Witten [22] have shown that Stacking works well when meta-instances are formed by probability distributions for each class instead of just a class label. A recent study [11] has shown that Stacking with Multi-Response Model Trees as the meta-level learning algorithm and probability distributions, is the most accurate heterogeneous classifier combination method of the Stacking family.

Selective Fusion [23,24] is a recent method for combining different classification algorithms that exhibits low computational complexity and high accuracy. It uses statistical procedures for the selection of the best subgroup among different classification algorithms and subsequently fuses the decision of the models in this subgroup with (Weighted) Voting.

3 Experimental Setup

This section provides information on the dataset, participating algorithms and combination methods that were used for the experiments.

3.1 Dataset

The protein classes considered in are the 10 most important protein families: PDOC00064 (a class of oxydoreductases), PDOC00154 (a class of isomerases), PDOC00224 (a class of cytokines and growth factors), PDOC00343 (a class of structural proteins), PDOC00561 (a class of receptors), PDOC00662 (a class of DNA or RNA associated proteins), PDOC00670 (a class of transferases), PDOC00791 (a class of protein secretion and chaperones), and PDOC50007 (a class of hydrolases). For clarity of presentation, the Prosite documentation ID, i.e. the PDOCxxxxx number was used to represent that class. Similarly, the Prosite access number i.e. the PSxxxxx was used to represent that motif pattern or profile. During the preprocessing, a training set was exported, consisting of 662 proteins that belong in barely 10 classes. Some proteins belonged in more than one class, thus the problem could be defined as a multi-label classification problem. The approach taken was to create separate classes in order to represent the classification of each multi-labeled protein. Thus, for a protein that belonged in two or more classes, a new class was created that was named after both the protein classes in which the protein belonged. This resulted to a total of 32 different classes. GenMiner [25] was used for the preparation of data.

3.2 Learning Algorithms and Combination Methods

We used 9 different learning algorithms at the base-level. These are general-purpose machine learning algorithms spanning several different learning paradigms (instance-based, rules, trees, statistical). They were obtained from the WEKA machine learning library [26], and used with default parameter settings unless otherwise stated:

- *DT*, the decision table algorithm of Kohavi [27].
- *JRip*, the RIPPER rule learning algorithm [28].
- *PART*, the PART rule learning algorithm [29].

- *J48*, the decision tree learning algorithm C4.5 [7], using Laplace smoothing for predicted probabilities.
- *IBk*, the *k* nearest neighbor algorithm [30].
- *K**, an instance-based learning algorithm with entropic distance measure [31].
- *NB*, the Naive Bayes algorithm [32] using the kernel density estimator rather than assume normal distributions for numeric attributes.
- *SMO*, the sequential minimal optimization algorithm for training a support vector classifier using polynomial kernels [33].
- *RBF*, WEKA implementation of an algorithm for training a radial basis function network [34].

The above algorithms were used alone and in conjunction with the following five different classifier combination methods: Stacking with Multi-Response Model Trees (SMT), Voting (V), Weighted Voting (WV), Evaluation and Selection (ES) and Selective Fusion (SF).

4 Results and Discussion

For the evaluation of the algorithms and combination methods, we used 10-fold stratified cross-validation. The original data set was split in 10 disjoint parts of approximately equal size and approximately equal class distribution. Each of these parts was sequentially used for testing and the union of the rest for training.

Table 1 presents the results concerning the mean error for each of the algorithms. We notice that the lowest error is exhibited by SMO followed closely by IBk, DT, J48, K* and PART. JRip is a bit worse, while NB and RBF exhibit quite low performance. The results verify the reputation of Support Vector Machines as a state-of-the-art classification method. Decision Trees (J48) and Instance-Based Learning methods (IBk, K*) also perform well, while Rule Based methods (JRip, PART) follow in performance.

Table 1. Mean error rate of the learning algorithms

	DT	JRip	PART	J48	IBk	K*	NB	SMO	RBF
Er.	0.024	0.035	0.026	0.024	0.023	0.024	0.610	0.021	0.739

After this evaluation the biologist might choose to use SMO for modeling the protein classification algorithm. However, the rest of the well-performing algorithms could also be used and might generalize better than SMO. In addition the combination of all these algorithms, or perhaps a subset of them could give better results. To investigate this issue we experimented with the combination methods that were mentioned in the precious section.

Table 2 presents the results concerning the mean error for each of the combination methods. ES simulates the process followed by someone that would like to use just the single best algorithm. 10-fold cross-validation is applied on the training set and the result of the evaluation guides the selection of the algorithm in the test set. This result, although good, it is worse than combining all the algorithms with Weighted Voting. This shows that the correction of uncorrelated errors through the voting process has helped in reducing the error rate. Simple Voting on the other hand did not perform well, due to the existence of bad performing models, such as NB and RBF. The state-of-the-art method of SMT performed very badly. The reason is that the large number of classes (32) leads to a very high dimensionality for the meta-level data set, which does not allow a good model to be induced. The best overall method is SF, which combines the best subset of the models that is selected using statistical tests. This result indicates that selection of the proper models and their combination can lead to very good results. It is worth noticing that SF selects and combines 6.3 models on average on the 10 folds. This result reinforces the previous conclusion that the combination of multiple algorithms results in error reduction, especially when coupled with a method for selecting the appropriate models.

Table 2. Mean error rate of the combination methods

	SB	V	VW	SF	SMT
Er.	0.024	0.195	0.021	0.019	0.558

5 Conclusions and Future Work

We have presented a comparative study of different classification algorithms and algorithm combination methods for the problem of motif-based protein classification. The results show that for a successful practical application of machine learning algorithms in such a real-world problem, one requires: a) a number of different classification algorithms, and b) a proper combination method that can automatically discard low performing models and combine the best models.

One of the issues that need to be investigated in the future is the approach taken for dealing with the multiple classes that each protein belongs to. This problem is common in biological domains and has not been considered extensively by the machine learning community. In the sequel of this work we intend to explore the effectiveness of alternative representations of the learning problem for the domain of protein classification.

References

1. Falquet, L., Pagni, M., Bucher, P., Hulo, N., Sigrist, C.J., Hofmann, K., Bairoch, A.: The PROSITE database, its status in 2002. Nucleic Acids Res. **30** (2002) 235–238
2. Bateman, A., Birney, E., Durbin, R., Eddy, S.R., Howem, K.L., Sonnhammer, E.L.L.: The Pfam protein families database. Nucleic Acids Res. **28** (2000) 263–266
3. Attwood, T.K., Croning, M.D.R., Flower, D.R., Lewis, A.P., Mabey, J.E., Scordis, P., Selley, J., Wright, W.: PRINT-S: the database formerly known as PRINTS. Nucleic Acids Res. **28** (2000) 225–227
4. Mitchell, T.: Machine Learning. McGraw Hill, New York (1997)

5. Baldi, P.F., Brunak, S.: Bioinformatics: The Machine Learning Approach. The MIT Press, Cambridge, MA (2001)
6. Wang, D., Wang, X., Honavar, V., Dobbs, D.: Data-driven generation of decision trees for motif-based assignment of protein sequences to functional families. In: Proceedings of the Atlantic Symposium on Computational Biology, Genome Information Systems & Technology (2001)
7. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufman, San Mateo (1993)
8. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press, New York (1995)
9. Duad, R., Hart, P.: Pattern Classification and Scene Analysis. Wiley, New York (1973)
10. Bairoch, A.: Prosite,: A dictionary of protein sites and patterns – User Manual. Swiss Institute of Bioinformatics, Geneva (1999)
11. Dzeroski, S., Zenko, B.: Is Combining Classifiers with Stacking Better than Selecting the Best One?. Machine Learning **54** (2004) 255–273
12. Brazdil, P.B., Soares, C., Da Costa, J.P.: Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. Machine Learning **50** (2003) 251-277
13. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.: Meta-learning by landmarking various learning algorithms. In: International Conference on Machine Learning, (2000)
14. Kalousis, A., Theoharis, T.: Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. In: Intelligent Data Analysis, (1999)
15. Bensusan, H., Giraud-Carrier, C., Kennedy, C.: A higher-order approach to meta-learning. In: ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, (2000)
16. Keller, J., Paterson I., Berrer, H.: An integrated concept for multi-criteria ranking of data mining algorithms. In: ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, (2000)
17. Giacinto, G., Roli, F.: Adaptive selection of image classifiers. In: Proceedings of the 9th International Conference on Image Analysis and Processing, (1997) 38-45
18. Woods, K., Kegelmeyer, W. P., Bowyer, K.: Combination of multiple classifiers using local accuracy estimates. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997) 405–410
19. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence **16** (1994) 66–75
20. Merz, C.J.: Dynamical selection of learning algorithms. In: Fisher, D., Lenz, H. J., eds: Learning from Data: Artificial Intelligence and Statistics, Springer-Verlag, (1995)
21. Wolpert, D.: Stacked generalization. Neural Networks **5** (1992) 241–259
22. Ting, K.M., Witten, I.H.: Issues in stacked generalization. Journal of Artificial Intelligence Research **10** (1999) 271–289
23. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective Voting of Heterogeneous Classifiers. In: Proceedings of the 15th European Conference on Machine Learning, Pisa, Italy (2004) 465–476
24. Tsoumakas, G., Angelis, L., Vlahavas, I.: Selective Fusion of Heterogeneous Classifiers. Intelligent Data Analysis **9** (2005) to appear.
25. Hatzidamianos, G., Diplaris, S., Athanasiadis, I., Mitkas, P.A.: GenMiner: A Data Mining Tool for Protein Analysis. In: Proceedings of the 9th Panhellenic Conference on Informatics, Thessaloniki, Greece (2003)
26. Witten, I., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann (1999)
27. Kohavi, R.: The power of decision tables. In: Proceedings of the 12th European Conference on Machine Learning (1995) 174–189

28. Cohen, W.: Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning (1995) 115–123
29. Witten, I., Frank, E.: Generating accurate rule sets without global optimization. In: Proceedings of the 15th International Conference on Machine Learning (1998) 144–151
30. Aha, D., Kibler, D., & Albert, M. Instance-based learning algorithms. Machine Learning **6** (1991) 37–66
31. Cleary, J., Trigg, L.: K*: An instance-based learner using an entropic distance measure. In: Proceedings of the 12th International Conference on Machine Learning (1995) 108–114
32. John, G., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (1995) 338–345
33. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Scholkopf, B., Burges, C., Smola, A, eds: Advances in Kernel Methods - Support Vector Learning, MIT Press (1998)
34. Bishop, C: Neural Networks for Pattern Recognition. Oxford University Press (1995)