To be published in:
The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 2003)

Protein Fold Class Prediction Using Neural Networks Reconsidered

Thomas Wiebringhaus[1,2], Ulrich Faigle[1,2], Dietmar Schomburg[2], Jutta Gebert[1,2], Christian Igel[3], Gerhard-Wilhelm Weber[1,2]

Keywords: protein fold class predictions, neural networks, support vector machines, generalization

1 Problem description.

Predicting the three-dimensional structure of a protein from its amino acid sequence is an important and challenging task in bioinformatics. In the following, we describe an application of feed-forward neural networks (NNs) for the protein fold classification task, introduce a tailored regularization method for sparse data, and compare the results to support vector machines (SVMs [9]).

We consider the problem of assigning a primary sequence to one of 42 fold classes. The data are taken from the *database for expected fold-classes* (DEF [7]). Each sequence is represented by its dipeptide-frequency without any additional associated physicochemical properties, i.e., the input dimension of the problem is 400. The data set is split into a set D_{APE} for training and a disjoint test set D_{TPE} as in [4, 6], consisting of 143 and 125 patterns, respectively.

2 Method.

A standard NN was employed for classification [2]. We did not reduce the input dimension (as done for the NNs in [4] by PCA), used 1-of-42 output encoding, and set *ad hoc* the number of hidden neurons to 10. This means, the network has by far more weights than training patterns are available. However, the generalization of NNs is surprisingly insensitive to excess capacity, see [1] and [3] for theoretical and empirical studies. *Early stopping* in conjunction with a suitable learning algorithm starting from small initial weights can "be used to stop training large nets when they have learned models similar to those learned by smaller nets of optimal size" [1]. Here, an improved version of the Rprop learning algorithm was used for training [5, 8].

We split D_{APE} randomly but ensuring an as equal distribution of the classes as possible into D_{train} and D_{validate} with 97 and 50 patterns, respectively. However, as we have only 147 training patterns for discriminating 42 classes, it is problematic to restrict the complete training process to a subset of D_{APE}, i.e., early stopping can not be applied in the standard way. Thus, we used a two-stage training process: First, the network is trained on D_{APE} until the classification error on D_{APE} vanished. Then training continued on D_{train} until generalization loss is observed on D_{validate}. The weight configuration yielding the smallest mean-squared-error on D_{validate} is taken as the final solution of the training process.

[1]Center for Applied Informatics Cologne, University of Cologne, 50931 Cologne, Germany
[2]Cologne University Bioinformatics Center, University of Cologne, 50674 Cologne, Germany
[3]Lehrstuhl für theoretische Biologie, Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany. E-mail: `christian.igel@neuroinformatik.ruhr-uni-bochum.de`, corresponding author

3 Results.

In [4], 11 different classification methods were applied to the described data sets including different discriminant analysis methods, nearest neighbor classification, projection pursuit regression, and NNs. SVMs with different kernels were used in [6] for comparison and gave the best results.

We trained our network starting from 10 small random weight initializations for 100 iterations. The apparent prediction error (APE, the classification error on D_{APE}) was 0 in all trials. The average classification error on the test set (TPE, test prediction error) was 22.96 % and the variance $\sigma^2 = 7.18 \cdot 10^{-5}$. Most trials including the one with the smallest mean-squared error on training and validation set (i.e., the one that would be selected) resulted in a network with a TPE of 22.4 %.

	SVM RBF	SVM POLY1	SVM POLY2	SVM POLY3	NN
APE	0	0	4.2	1.4	0
TPE	23.2	28.8	32	32.8	**22.4**

Table 1: Results using the neural network (NN) compared to SVMs with different kernels [6], a radial basis kernel (RBF) and polynomial kernels of degrees 1, 2, and 3 (POLY1, POLY2, POLY3).

In our (preliminary) experiments, NNs combined with an appropriate regularization scheme showed better classification results than reported for SVMs and other classification methods [4, 6], see Tab. 1. Note that it is sufficient to train a single NN for a few generations, but that 42 SVMs are needed for the *one-against-all* multiple-class discrimination approach in [6]. The proposed method to deal with sparse data is not limited to protein fold classification, but may be applicable to a wide range of bioinformatics tasks.

References

[1] P. L. Bartlett. For valid generalization the size of the weights is more important than the size of the network. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 134. The MIT Press, 1997.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[3] R. Caruana, S. Lawrence, and C. L. Giles. Overfitting in neural networks: Backpropagation, conjugate gradient, and early stopping. In *Advances in Neural Information Processing Systems 13*, pages 402–408, Denver, Colorado, 2001. MIT Press.

[4] L. Edler, J. Grassmann, and S. Suhai. Role and results of statistical methods in protein fold class prediction. *Mathematical and Computer Modelling*, 33:1401–1417, 2001.

[5] C. Igel and M. Hüsken. Empirical evaluation of the improved Rprop learning algorithm. *Neurocomputing*, 50(C):105–123, 2003.

[6] F. Markowetz. Support vector machines in bioinformatics. Master's thesis, Mathematics Department of the University of Heidelberg, 69120 Heidelberg, Germany, 2001.

[7] M. Reczko and H. Bohr. The DEF data base of sequence based protein fold class predictions. *Nucleic Acids Research*, 22(17):3616–3619, 1994.

[8] M. Riedmiller. Advanced supervised learning in multi-layer perceptrons – From backpropagation to adaptive learning algorithms. *Computer Standards and Interfaces*, 16(5):265–278, 1994.

[9] V. N. Vapnik. *Statistical Learning Theory.* Wiley, New-York, 1998.