Halmstad University Post-Print

Almost Linear Biobasis Function Neural Networks

Liwen You and Thorsteinn Rögnvaldsson

N.B.: When citing this work, cite the original article.

©2007 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

You L, Rögnvaldsson T. Almost Linear Biobasis Function Neural Networks. In: The 2007 International Joint Conference on Neural Networks: IJCNN 2007 conference proceedings : August 12-17, 2007, Renaissance Orlando Resort, Orlando, Florida, USA. Piscataway, N.J.: IEEE; 2007. p. 1774-1778. International Conference on Neural Networks, 2007.

DOI: <http://dx.doi.org/10.1109/IJCNN.2007.4371226>

Copyright: IEEE

Post-Print available at: Halmstad University DiVA
<http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-5021>

Almost Linear Biobasis Function Neural Networks

Liwen You and Thorsteinn Rögnvaldsson

Abstract—An analysis of biobasis function neural networks is presented, which shows that the similarity metric used is a linear function and that bio-basis function neural networks therefore often end up being just linear classifiers in high dimensional spaces. This is a consequence of four things: the linearity of the distance measure, the normalization of the distance measure, the recommended default values of the parameters, and that biological data sets are sparse.

I. INTRODUCTION

Biobasis function neural networks (BBFNN) were introduced by Thomson and Yang in 2002 [1]. They were first used for predicting proteolytic cleavage by trypsin and HIV-1 protease but have since been used several times also for other studies within the protein domain. Biobasis function neural networks are similar to radial basis function neural networks (RBFNN) but the standard Euclidean, or Mahalanobis, distance is replaced by a biological similarity distance based on mutation probabilities. The basis functions in BBFNN have the general form

$$\phi(\mathbf{P}) = \exp \left[\alpha \frac{s(\mathbf{P}, \mathbf{Q}) - \beta}{\beta} \right], \quad (1)$$

where $\mathbf{P}$ is a query sequence (amino acid sequence) and $\mathbf{Q}$ a reference sequence associated with the basis function, similar to the center point for basis functions in RPFNN. The function $s(\mathbf{P}, \mathbf{Q})$ is the biological similarity function (defined later in this paper). The parameters α and β are fixed, i.e. not adjusted during training. The training of the BBFNN is done in the same way as an RBFNN model is trained, i.e. using a least squares algorithm.

The original BBFNN model [1] [2] used all training sequences as biobases (reference sequences $\mathbf{Q}$). This produced large models with many parameters and a reduced version was therefore introduced by Berry *et al.* [3]. The reduced version uses genetic algorithms or mutual information methods to search for an optimal basis set that is significantly smaller than the full training set; Berry *et al.* [3] report being able to reduce the basis set to about one third of the full training set.

A central element in the BBFNN is the similarity measure, $s(\mathbf{P}, \mathbf{Q})$, which builds on a mutation matrix [4]. The choice of this mutation matrix affects the prediction result; Yang and Thomson [5] have compared the model performance of different mutation matrices and conclude that different mutation matrices give quite different prediction performance and that it is not possible to tell a priori which one that will work best.

Liwen You and Thorsteinn Rögnvaldsson are with the School of Information Science, Computer and Electrical Engineering, Halmstad University, Sweden (email: liwen.you@ide.hh.se; denni@ide.hh.se).

We have also tried biobasis functions in our research, e.g. on predicting cleavage by HIV-1 protease [6]. We have, however, been unable to get outstanding, or even better, performance from biobasis functions compared with other methods. An example is in Figure 1, which shows the prediction accuracy performance of different algorithms (a simple perceptron, a linear support vector machine and the BBFNN) when predicting cleavage by the HIV-1 protease on a standard 362 octamer data set. Here, biobasis functions did not do better, rather worse, than a simple linear perceptron or a linear support vector machine. These results have been puzzling, even disturbing, indicating that we have been unable to get the full potential out of the biobasis functions. However, we have analyzed the biobasis functions, an analysis that we present in this paper, and come to the conclusion that we should not expect them to be better than linear classifiers unless we use a very large number of basis sequences or parameter settings that are very different from the recommended default values.

Fig. 1. Out-of-sample prediction performance on 362 octamers as a function of training set size (BBF = bio-basis function with Blosom62 substitution matrix and $\alpha = 1$, LSVM = linear support vector machine, SP = linear simple perceptron). For LSVM and SP, octamers were coded by using sparse coding.

II. THEORY

A. Sparse coding and linear classifiers

A N -mer, a sequence with N amino acids, is denoted by

$$\mathbf{P} = P_1 P_2 \dots P_N,$$

where P_i can take on any one of the amino acid values, e.g. { A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y,

V}. A standard procedure is to use a sparse coding for N -mers [7]. Each amino acid is, in the case of 20 amino acids, represented by a 20 bit vector with 19 bits set to zero and one bit set to one. This transformation, $\mathbf{P} \rightarrow \mathbf{X}(\mathbf{P})$, maps each N -mer $\mathbf{P}$ to one of the corners $\mathbf{X}$ of a $20 \times N$ dimensional hypercube (or, from another point of view, all the N -mers lie on the surface of a $20 \times N$ dimensional sphere).

Furthermore, the vector $\mathbf{X}$ fulfils the N linear constraints

$$\mathbf{X}^T \mathbf{1}_{((i-1) \cdot 20 + 1) - (i \cdot 20)} = 1, i = 1, \dots, N \quad (2)$$

where $\mathbf{1}_{i-j}$ is a $20 \times N$ -dimensional binary vector with positions i through j set to one and the remaining set to zero. These N constraint equations reduce the effective dimensionality of the problem from $20 \times N$ down to $19 \times N$ (i.e. all the N -mers lie in a $19 \times N$ dimensional subspace).

We use the term *linear classifier* to denote a sequence analysis algorithm that maps the N -mer in sparse representation to a desired output y in the following way

$$y(\mathbf{P}) = \Theta[w_0 + \mathbf{W}^T \mathbf{X}(\mathbf{P})] = \Theta \left[w_0 + \sum_{k=1}^K w_k x_k(\mathbf{P}) \right] \quad (3)$$

where the transfer function Θ may be a nonlinear function (e.g. a step function or a sigmoid function). Examples of linear classifiers are the simple perceptron and the logistic regression model.

B. Substitution matrix and similarity distance

Substitution matrices are derived by extracting similarity distances between amino acids from specific databases, which represent special domain information [4]. A substitution matrix, e.g. Blosom50, can be written as

$$\mathbf{M} = \begin{pmatrix} M_{AA} & M_{AR} & M_{AN} & \dots & M_{AV} \\ M_{RA} & M_{RR} & M_{RN} & \dots & M_{RV} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ M_{VA} & M_{VR} & M_{VN} & \dots & M_{VV} \end{pmatrix},$$

where each element represents an amino acid pair distance, which is a numerical value. Substitution matrices are symmetric, i.e. $\mathbf{M}^T = \mathbf{M}$.

The similarity distance $s(\mathbf{P}, \mathbf{Q})$, measured with the substitution matrix $\mathbf{M}$, between two N -mers $\mathbf{P} = P_1 P_2 \dots P_N$ and $\mathbf{Q} = Q_1 Q_2 \dots Q_N$ is defined as:

$$s(\mathbf{P}, \mathbf{Q}) \equiv \sum_{i=1}^N M_{P_i Q_i}. \quad (4)$$

If we define the $20N \times 20N$ block diagonal matrix $\mathbf{M}_N$

$$\mathbf{M}_N \equiv \begin{pmatrix} \begin{bmatrix} \mathbf{M} \end{bmatrix} & 0 & \dots & 0 \\ 0 & \begin{bmatrix} \mathbf{M} \end{bmatrix} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \begin{bmatrix} \mathbf{M} \end{bmatrix} \end{pmatrix} \quad (5)$$

i.e. the matrix where the substitution matrix $\mathbf{M}$ is repeated N times along the diagonal and all other elements are zero, then we can write the similarity distance (4) as

$$s(\mathbf{P}, \mathbf{Q}) = \mathbf{X}^T(\mathbf{P}) \mathbf{M}_N \mathbf{X}(\mathbf{Q}) \quad (6)$$

where $\mathbf{X}(\mathbf{P})$ and $\mathbf{X}(\mathbf{Q})$ are the sparse notation representations of the N -mers $\mathbf{P}$ and $\mathbf{Q}$. The substitution matrix $\mathbf{M}$ is symmetric and the similarity distance is therefore also symmetric; $s(\mathbf{P}, \mathbf{Q}) = s(\mathbf{Q}, \mathbf{P})$.

The maximum value for the similarity measure is the self-similarity, i.e.

$$\max_{\mathbf{P}} s(\mathbf{P}, \mathbf{Q}) = s(\mathbf{Q}, \mathbf{Q}) = \mathbf{X}^T(\mathbf{Q}) \mathbf{M}_N \mathbf{X}(\mathbf{Q}). \quad (7)$$

There are some important messages in expressions (5) and (6). The block diagonal form of the matrix $\mathbf{M}_N$ shows that each position in the N -mer is treated independently of the other positions. Furthermore, each position is treated the same as any other position since the same matrix $\mathbf{M}$ is repeated along the diagonal. This means that information like “position i is unimportant” is not easily handled with the similarity metric. A way to incorporate this would be to have a weighting factor for each matrix $\mathbf{M}$.

C. Biobasis functions

Biobasis function neural networks are of the form

$$\begin{aligned} y(\mathbf{P}) &= w_0 + \sum_{k=1}^K w_k \exp \left[\alpha \frac{s(\mathbf{P}, \mathbf{B}_k) - \beta_k}{\beta_k} \right] \\ &= w_0 + \sum_{k=1}^K w_k e^{-\alpha} \exp \left[\alpha \frac{s(\mathbf{P}, \mathbf{B}_k)}{\beta_k} \right], \end{aligned} \quad (8)$$

where $\mathbf{B}_k$ is the k :th base N -mer, $s(\mathbf{P}, \mathbf{B}_k)$ is the similarity distance between N -mer $\mathbf{P}$ and the base N -mer $\mathbf{B}_k$ measured using a biological substitution matrix, like Dayhoff or Blosom62; and β_k is the maximum similarity measurement associated with the k :th basis, i.e. $\beta_k = s(\mathbf{B}_k, \mathbf{B}_k)$. This means that $0 < s(\mathbf{P}, \mathbf{B}_k)/\beta_k \leq 1$.

Biobasis functions are fitted by first selecting a set of base N -mers, $\mathbf{B}_k$, and then fitting the parameters w_k using, e.g., linear least squares. For the reduced BBFNN is a search done over different sets of base N -mers using, e.g., genetic algorithms. The scaling parameter α is fixed. Thomson *et al.* state [2] that the biobasis model accuracy is insensitive to α over a wide range of values and $\alpha = 1$ in the studies.

A series expansion of expression (8) yields

$$\begin{aligned}
y(\mathbf{P}) &= w_0 + \sum_{k=1}^K w_k e^{-\alpha} \exp \left[\alpha \frac{s(\mathbf{P}, \mathbf{B}_k)}{\beta_k} \right] \\
&= w_0 + \sum_{k=1}^K w_k e^{-\alpha} + \alpha \sum_{k=1}^K w_k e^{-\alpha} \frac{s(\mathbf{P}, \mathbf{B}_k)}{\beta_k} + \\
&\quad \frac{\alpha^2}{2} \sum_{k=1}^K w_k e^{-\alpha} \frac{s^2(\mathbf{P}, \mathbf{B}_k)}{\beta_k^2} + \dots
\end{aligned} \tag{9}$$

And if we combine (9) with expression (6) we see that the biobasis function can be written as

$$y = A + \mathbf{B}^T \mathbf{X} + \mathbf{X}^T \mathbf{C} \mathbf{X} + \dots$$

where A is a scalar, $\mathbf{B}$ is a vector, $\mathbf{C}$ is a matrix, and so on and so forth (the $\mathbf{P}$ is dropped for notational convenience). Also, the smaller α is, the more linear does the biobasis function become.

However, the dependence on α is not the whole story. The biobasis functions are typically used to dichotomize a data set and the boundary between the two parts of the data set will typically fall at sequences that are not very similar to any of the base sequences. The sequence similarity divided by the maximum similarity, $s(\mathbf{B}_k, \mathbf{P})/\beta_k$, will therefore be very small for sequences on the boundary. Hence, the biobasis function will be most close to linear on the decision boundary (the boundary that separates the two parts of the dichotomy), which is where it matters. It is therefore very likely that a biobasis function used for classification effectively ends up being a linear classifier, like expression (3).

III. EXPERIMENTS

The structure of the sparse orthogonal data space, with the patterns placed at the corners in a hypercube, makes it difficult to prove analytically that a specific classification approach is equivalent to a linear classifier. This is because a model that is slightly nonlinear may very well be practically equivalent to a linear model, since the curvature of the decision boundary within the hypercube does not matter. An empirical approach was therefore chosen to check whether the biobasis models were linear or nonlinear classifiers. The approach was to draw many samples from the biobasis function and check whether these samples were linearly separable or not. The experiments were done in the following way:

1. A random BBFNN model for a two-class problem was created with K base sequences. Random here means that the base sequences were created randomly and assigned to one of two categories randomly. The model was balanced, i.e. there were equal numbers of base sequences belonging to both categories.
2. This model was then used to label a large data set of random sequences.
3. A linear support vector machine was then used to test if the labeled data set was linearly separable. If it was,

then we concluded that the BBFNN model had produced a linear decision.

These steps (1-3) were repeated several times, with varying α , K and size of the generated data sets. The Blosom50 substitution matrix was used throughout.

The notion was that if the labeled data set with many sequences was linearly separable, then the biobasis function that generated the data set was probably also very close to linear.

The linear support vector machine (LSVM) sometimes fails to find a linear separation when one exists, so the estimated probability for linearity is just a lower limit. However, the LSVM is much faster than the simple perceptron (which is guaranteed to find a solution if one exists) so that the experiments can be carried out within a reasonable time.

A different set of experiments was also done to test the effect of the similarity distance, without using the exponential function, so that the effect of the exponential function could be separated from the effect of the chosen metric. This second set of experiments was done as follows:

1. A set of K random base sequences was drawn (i.e. the base sequences were created randomly and assigned to one of two categories randomly). The set was balanced, i.e. there were equal numbers of base sequences belonging to both categories.
2. A set of random sequences was then drawn and labeled according to how close each sequence was, as measured with a Blosom50 similarity matrix, to the base sequences. Each sequence was labeled with the same label as its closest base sequence.
3. A simple linear perceptron was then used to test if the labeled data set was linearly separable.

Ten experiments were done for each value of the size of the data set and the number of biobasis sequences. Octamers ($N = 8$) were used throughout this second experiment set. The intention with this second set of experiments was to test if there was any effect associated with using a similarity metric and few basis sequences. This latter set of experiments correspond to labeling new sequences based on a nearest neighbor algorithm and a similarity metric measure.

IV. RESULTS

Figure 2 shows the result of some experiments when $N = 8$ (octamers). The z-axis denotes the average correct classification rates of LSVM models over 10 runs with $C = 1000$, when a new random data set is selected every run (LibSVM Matlab Toolbox version 2.83 was used to build LSVM models [8]). A value equal to one means that the generated data set is linearly separable. The x-axis shows the $\log_2$ of the number of basis functions, i.e. 9 corresponds to $2^9 = 512$ basis functions and 10 corresponds to $2^{10} = 1024$ basis functions. The y-axis shows the value of α .

When the data set is small (300 samples) then it is always linearly separable. This is to be expected since essentially all data sets with a size below the capacity for the linear classifier are linearly separable [9]; the capacity for a linear

Fig. 2. Effects of α and number of basis on the complexity of biobasis functions. The z-axis denotes the average correct classification rates of LSVM models over 10 runs with $C=1000$ on the generated peptides from biobasis functions.

classifier using N -mers is twice the effective dimension, i.e. 304 ($19 \times 8 \times 2$) for octamers. However, the probability for a data set to be linearly separable decreases very quickly above the capacity limit. The probability for a data set of size M to be linearly separable in a high-dimensional space of dimension D is approximately [9]

$$P(\text{linear}) \approx 0.5 \left[1 + \text{erf} \left(\sqrt{M/2} (2D/M - 1) \right) \right] \quad (10)$$

where erf denotes the error function. The probability for a balanced data set of size M to be linearly separable is even less. Thus, a balanced data set with 1000 random octamer samples is very unlikely to be linearly separable.

The experiments show, however, that even large data samples generated with biobasis functions are linearly separable when α is small. It is not until the number of basis sequences becomes very large, $2^8 = 256$, that the probability for linear separability drops significantly.

The results of the second set of experiments, those using a nearest neighbor approach with Blosom50 matrix, are shown in Figure 3. When reading this figure it is useful to relate to expression (10). The effective dimensionality is 152 (19×8) so we expect the transition from linear to nonlinear to happen at $2 \times 152 = 304$ sequences; many data sets with less than 304 sequences will be linearly separable, whereas data sets with more than 304 sequences will tend not be linearly separable. The almost linear nature of the biobasis functions (first experiment set) is not reflected in the similarity metric

(second experiment set). Even with a small number of basis sequences is the behavior as if the resulting cluster function is nonlinear (i.e. there is a transition from linear to nonlinear at approximately 300 sequences).

Fig. 3. The results of the nearest neighbor approach when using Blosom50 similarity matrix. The z-axis denotes the probability for the data set to be nonlinear (i.e. one means nonlinear data set and zero means linear data set).

V. CONCLUSION

It was demonstrated that the separation power of the recently introduced biobasis functions is often not very much more than that of linear classifiers, if biobasis functions are used in the way they have been used so far, with a moderate number of basis functions and a scale factor $\alpha = 1$. The analysis also shows that this is not a consequence of which substitution matrix that is used. The linearity is a consequence of four things: the linearity of the distance metric, the small value for the scale factor α , the normalization of the similarity distance with the maximum similarity β , and the use of a few (a few hundred) basis sequences. A set of experiments using a nearest neighbor algorithm and similarity distance matrices produce nonlinear decision boundaries with even a small number of basis sequences, which demonstrated clearly that the linearity is not a consequence of the metric alone.

In addition, biobasis functions do not show advantages on computation time in that it takes about 18.8 seconds for biobasis functions to run the experiment once shown in Figure 1, compared to 2.3 seconds by linear support vector machines when using the same computer to test them.

In conclusion, biobasis functions that operate on N -mers and with a reasonable number of basis sequences (e.g. a few hundred) are probably not doing anything more advanced than a linear separation in the sparse orthogonal space.

REFERENCES

- [1] R. Thomson and Z. R. Yang, "A novel basis function neural network," in *Proc. of 9th International Conference on Neural Information processing (ICONIP '02)*, L. Wang, J. C. Rajapakse, K. Fukushima, S.-Y. Lee, and X. Yao, Eds. IEEE press, 2002, pp. 441–446.

- [2] R. Thomson, T. C. Hodgman, Z. R. Yang, and A. K. Doyle, "Characterizing proteolytic cleavage site activity using bio-basis function neural networks," *Bioinformatics*, vol. 19, pp. 1741–1747, 2003.
- [3] E. A. Berry, A. R. Dalby, and Z. R. Yang, "Reduced bio basis function neural network for identification of protein phosphorylation sites: comparison with pattern recognition algorithms," *Computational Biology and Chemistry*, vol. 28, pp. 75–85, 2004.
- [4] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proc. Natl. Acad. Sci.*, vol. 89, pp. 10 915–10 919, 1992.
- [5] Z. R. Yang, , and R. Thomson, "Bio-basis function neural network for prediction of protease cleavage sites in proteins," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 263–274, 2005.
- [6] L. You, D. Garwicz, and T. Rgnvaldsson, "Comprehensive bioinformatic analysis of the specificity of human immunodeficiency virus type 1 protease," *J. Virol.*, vol. 79, pp. 12 477–86, 2005.
- [7] N. Qian and T. J. Sejnowskij, "Predicting the secondary structure of globular proteins using neural network models," *Journal of Molecular Biology*, vol. 202, pp. 865–884, 1988.
- [8] C. C. Chang and C. J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [9] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*, ser. Lecture Notes, Santa Fe Institute, Studies in the Science of Complexity. Addison-Wesley, 1991, vol. 1.