USING REAL-VALUED META CLASSIFIERS TO INTEGRATE BINDING SITE PREDICTIONS

Yi Sun, Mark Robinson, Rod Adams, Paul Kaye, Alistair G. Rust, Neil Davey
University of Hertfordshire, College Lane, Hatfield, Hertfordshire AL10 9AB
comrys, m.robinson, r.g.adams, p.h.kaye, n.davey@herts.ac.uk, arust@systemsbiology.org

***Abstract*— Currently the best algorithms for transcription factor binding site prediction are severely limited in accuracy. There is good reason to believe that predictions from these different classes of algorithms could be used in conjunction to improve the quality of predictions. In this paper, we apply single layer networks, rules sets and support vector machines on predictions from 12 key real valued algorithms. Furthermore, we use a 'window' of consecutive results in the input vector in order to contextualise the neighbouring results. We improve the classification result with the aid of under- and over- sampling techniques. We find that support vector machines outperform each of the original individual algorithms and the other classifiers employed in this work. In particular they have a better tradeoff between recall and precision.**

I. INTRODUCTION

In this paper, we address the problem of identifying transcription factor binding sites on sequences of DNA. There are many different algorithms in current use to search for binding sites [22], [3], [23], [5]. However, most of them produce a high rate of false positive predictions. The problem addressed here is to reduce these false positive predictions by means of classification techniques taken from the field of machine learning.

To do this we first integrate the results from 12 different base algorithms for identifying binding sites, using non-linear classification techniques. To further improve classification results, we employ windowed inputs, where a fixed number of consecutive results are used as an input vector, so as to contextualise the neighbouring results. The data has two classes labeled as either binding sites or non-binding sites, with about 93% used being non-binding sites. We make use of sampling techniques, working with a traditional neural network: single layer networks (SLN), rules sets (C4.5-Rules) and a contemporary classification algorithm: support vector machines (SVM).

In previous work we have used binary valued base algorithms [19], here we extend this to use as much information as possible as provided by the real valued base algorithms.

We expound the problem domain in the next section. In Section III, we introduce the datasets used in this paper. We explain how we apply under- and over- sampling techniques in Section IV. A set of common metrics and receiver operating characteristics graphs for assessing classifier performance are covered in Section V. Section VI briefly introduces our experiments and Section VII gives all the experimental results. The paper ends in Section VIII with conclusions.

II. PROBLEM DOMAIN

One of the most exciting and active areas of research in biology currently, is understanding how the exquisitely fine resolution of gene expression is achieved at the molecular level. It is clear that this is a highly non-trivial problem. While the mapping between the coding region of a gene and its protein product is straightforward and relatively well understood, the mapping between a gene's expression profile and the information contained in its non-coding region is neither so simple, nor well understood at present. It is estimated that as much as 50% of the human genome is cis-regulatory DNA [15] , undeciphered for the most part and tantalisingly full of regulatory instructions. A cis-regulatory component consists of DNA that encodes a site for protein-DNA interaction in a gene regulatory system, conversely, a trans-regulatory component consists of a protein that binds to a cis-regulatory DNA sequence. Cis-regulatory elements form the nodes connecting the genes in the regulatory networks, controlling many important biological phenomena, and as such are an essential focus of research in this field [2]. Lines of research likely to directly benefit from more effective means of elucidating the cis-regulatory logic of genes include embryology, cancer and the pharmaceutical industry.

It is known that many of the mechanisms of gene regulation act directly at the transcriptional or sequence level, for example in those genes known to play integral roles during embryogenesis [2]. One set of regulatory interactions are those between a class of DNA-binding proteins known as transcription factors and short sequences of DNA which are bound by the proteins by

virtue of their three dimensional conformation. Transcription factors will bind to a number of different but related sequences. A base substitution in a cis-regulatory element will commonly, simply modify the intensity of the protein-DNA interaction rather than abolish it. This flexibility ensures that cis-regulatory elements, and the networks in which they form the connecting nodes, are fairly robust to various mutations. Unfortunately, it complicates the problem of predicting the cis-regulatory elements from out of the random background of the non-coding DNA sequences.

The current state of the art algorithms for transcription factor binding site prediction are, in spite of recent advances, still severely limited in accuracy. We show that in a large sample of annotated yeast promoter sequences, a selection of 12 key algorithms were unable to reduce the false positive predictions below 80%, with between 20% and 65% of annotated binding sites recovered. These algorithms represent a wide variety of approaches to the problem of transcription factor binding site prediction, such as the use of regular expression searches, PWM scanning, statistical analysis, co-regulation and evolutionary comparisons. There is however good reason to believe that the predictions from these different classes of algorithms are complementary and could be integrated to improve the quality of predictions.

In the work described here we take the results from the 12 aforemention algorithms and combine them into 2 different feature vectors, as shown in next section. We then investigate whether the integrated classification results of the algorithms can produce better classifications than any one algorithm alone.

III. Description of The Data

The data has 68910 possible binding positions and a prediction result for each of the 12 algorithms. The 12 algorithms can be categorised into higher order groups as Single sequence algorithms (7) [22], [1], [17], [20]; Coregulatory algorithms (3) [3], [11]; A Comparative algorithm (1) [23]; An Evolutionary algorithm (1) [5].

The label information contains the best information we have been able to gather for the location of known binding sites in the sequences. We use -1 to denote the prediction that there is no binding site at this location and +1 to denote the predictions that there is a binding site at this location. For each of the base 12 algorithms, a prediction result can be either binary or real valued, see Figure 1. The data therefore consists of 68910 12-ary real vectors each with an associated binary value.

In this work, we divide our dataset into a training set and a test set: the first $2/3$ is the training set and the last $1/3$ is the test set. Amongst the data there are repeated vectors, some with the same label (repeated items) and some with different labels (inconsistent items). It is

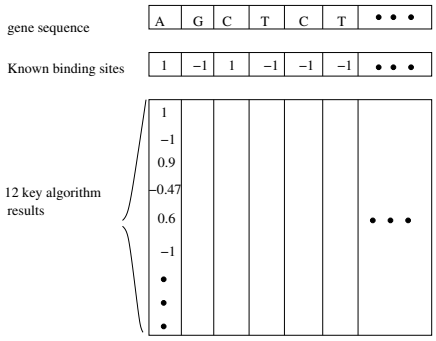

Fig. 1. The dataset has 68910 columns, each with a possible binding prediction (binary or real value). The 12 algorithms give their own prediction for each sequence position and one such column is shown.

obviously unhelpful to have these repeated or inconsistent items in the training set, so they are removed. We call the resulting data the consistent training set. However in the case of the test set we consider both the full set of data and the subset consisting of only the consistent test items. As can be seen in Table 1, the removal of repeated and inconsistent data dramatically reduces the number of data items: roughly 70% of data is lost.

As the data is drawn from a sequence of DNA nucleotides the label of other near locations is relevant to the label of a particular location. We therefore contextualise the training and test data by windowing the vectors as shown in Figure 2. We use the locations up to three either side, giving a window size of 7, and a consequent input vector size of 84. This has the considerable additional benefit of eliminating most of the repeated and inconsistent data: as can be seen in Table 1 now less than 5% of the data is lost.

Table 1 gives the sizes of all the different data sets used in this paper. The training set consists of either single vectors or windowed vectors. In both cases only consistent, non-repeating data is used. The test data consists of either single vectors or windowed vectors as appropriate. Either the full test set or the relevant consistent subset is used. There is however, a special case, namely when we want to compare the windowed model with the single input version. Here we want to evaluate the windowed model on the locations represented in the consistent test set of the single vector model. We therefore construct a test set for the windowed model consisting of only those vectors corresponding to the 7 locations around each of the data points in the single consistent test set.

IV. Sampling Techniques for Imbalanced Dataset Learning

In our dataset, there are less than 10% binding positions amongst all the vectors, so this is an *imbalanced*

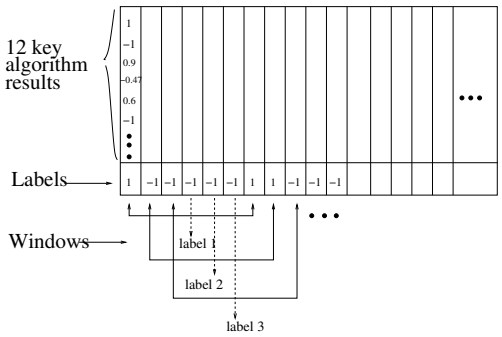

Fig. 2. The window size is set to 7 in this study. The middle label of 7 continuous prediction sites is the label for a new windowed inputs. The length of each windowed input now is 12×7.

TABLE I

DESCRIPTION OF THE DATASETS USED IN THIS WORK.

type			size
training	consistent	single	12790
		windowed	42919
test	consistent	single	5966
		restricted windowed	5966
	full	single	22967
		windowed	22967

dataset [13]. Since the dataset is imbalanced, the supervised classification algorithms will be expected to over predict the majority class, namely the non-binding site category. There are various methods of dealing with *imbalanced* data [12]. In this work, we concentrate on the data-based method [7]: using under-sampling of the majority class (negative examples) and over-sampling of the minority class (positive examples). We combine both over-sampling and under-sampling methods in our experiments.

For under-sampling, we randomly selected a subset of data points from the majority class. The over-sampling case is more complex. In [13], the author addresses an important issue that the class imbalance problem is only a problem when the minority class contains very small subclusters. This indicates that simply over sampling with replacements may not significantly improve minority class recognition. To overcome this problem, we apply a synthetic minority over-sampling technique as proposed in [7]. For each pattern in the minority class, we search for its $K-$nearest neighbours in the minority class using Euclidean distance. Since the dataset is a mixed one of continuous and binary features, we follow the suggestion in [7]: when the binary features differ between a pattern and its nearest neighbours, then the median of standard deviations of all continuous features for the minority class is included in the Euclidean distance. A new pattern belonging to the minority class can then be generated as follows: for continuous features, the difference of each feature between the pattern and its nearest neighbour is taken, and then multiplied by a random number between 0 and 1, and added to the corresponding feature of the pattern; for binary features, the majority voting principle to each element of the $K-$nearest neighbours in the feature vector space is employed. We take 5 nearest neighbours, and double the number of items in the minority class. The actual ratio of minority to majority class is determined by the under-sampling rate of the majority class. According to our previous experience, we set the final ratio to a half, which works well in this work.

V. CLASSIFIER PERFORMANCE

It is apparent that for a problem domain with an imbalanced dataset, classification accuracy rate is not sufficient as a standard performance measure. To evaluate classifiers used in this work, we apply Receiver Operating Characteristics (ROC) analysis [8], and several common performance metrics, such as recall, precision and F-score [6], [14], which are calculated to understand the performance of the classification algorithm on the minority class. Prior to introducing ROC curves, we give definitions of several common performance metrics.

A. Performance metrics

Based on the confusion matrix computed from the test results, several common performance metrics can be defined as in Table , where TN is the number of true negative samples; FP is false positive samples; FN is false negative samples; FP is true positive samples.

TABLE II

PERFORMANCE METRICS

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}), \qquad \text{Precision} = \text{TP} / (\text{TP} + \text{FP}),$$
$$\text{F-score} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}, \qquad \text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FN+TN+FP}},$$
$$\text{fp\_rate} = \frac{\text{FP}}{\text{FP+TN}}.$$

B. ROC curves

ROC analysis has been used in the field of signal detection for a long time. Recently, it has also been employed in the machine learning and data mining domains. Here we follow [8] to give a basic idea of ROC curves.

1) ROC curves: In a ROC diagram, the *true positive rate* (also called recall) is plotted on the Y axis and the *false positive rate* (fp_rate) is plotted on the X axis. Points in the top left of the diagram therefore have a high TP rate and a low FP rate, and so represent good classifiers. The classifiers used here all produce a real valued output, that can be considered as a class

membership probability. It is normal when using a ROC diagram to compare classifiers, to generate a set of points in ROC space by varying the threshold used to determine class membership. In this way a ROC curve corresponding to the performance of a single classifier, but with a varying threshold, is produced. One classifier is clearly better only when it dominates another over the entire performance space [8]. One attractive property of ROC curves is that they are insensitive to changes in class distribution, which makes them useful for analysing performance of classifiers using imbalanced datasets.

As noted for a ROC curve to be generated a real valued classifier is needed. The original SVM is a binary classifier. As described in [21] it is possible for the SVM to generate probabilistic outputs. For majority voting and weighted majority voting, we adopt methods proposed in [9]. The score assigned to each pattern is the fraction of votes won by the majority in majority voting; while in weighted majority voting, each base algorithm votes with its *confidence*, which is measured by the probability that the given pattern (**I**) is positive (**P**), i.e., $p(\mathbf{P}|\mathbf{I}) \approx$ TP / (TP + FP). The class with the highest summed confidence wins, and the score is the average confidence. For the neural network classifiers a real valued output is automatically generated.

Often to measure a classifier performance, it is convenient to use a single metric and the area under a ROC curve (AUC) can be used for this purpose. Its value ranges from 0 to 1. An effective classifier should have an AUC more than 0.5.

VI. Experiments

The classification techniques used in this work are single layer network (SLN) [4], support vector machine (SVM) [18], rule sets (C4.5-Rules) [16], majority voting (MV), and weighted majority voting (WMV).

The SVM experiments were completed using LIBSVM, which is available from the URL
`http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
The C4.5-Rules experiments were done using C4.5 software from [16]. C4.5-Rules is a companion program to C4.5. It creates rules sets by post-processing decision trees generated using C4.5 algorithm first. The others were implemented using the NETLAB toolbox, which is available from the URL
`http://www.ncrg.aston.ac.uk/netlab/`.

A. Parameter Settings

Since we do not have enough data to build up an independent validation set to evaluate the model, all the user-chosen parameters are obtained using cross-validation. There are two training sets (single or windowed), and for each of these sets, and each classifier, the following cross validation procedure is carried out. The training set is divided into 5 equal subsets, one of which is to be a validation set, and there are therefore 5 possible such sets. For each classifier a range of reasonable parameter settings are selected. Each parameter setting is validated on each the five validation sets, having previously been trained on the other $4/5$ of the training data. The mean performance, as measured by the AUC metric over these 5 validations, is taken as the overall performance of the classifier with this parameter setting. The parameter setting with best performance is then used with this classifier and the corresponding data set (single or windowed) in the subsequent experiments. For example the SVM has two parameters and six different combinations were evaluated.

There are several approaches to generate an averaging ROC curve from different test sets [8]. In this paper, average ROC curves of the cross-validation results are obtained by first generating a ROC curve for each of the validation sets, and then by calculating the average scores from them.

The standard deviation of the AUC can therefore be attained either using the cross-validation method, or when only a single curve is available, approximated as follows [10], $se = \sqrt{\frac{A(1-A)+(N_p-1)(Q_1-A^2)+(N_n-1)(Q_2-A^2)}{N_n N_p}}$, where A denotes AUC, N_n and N_p are the number of negative and positive examples respectively, and $Q_1 = \frac{A}{2-A}$, $Q_2 = \frac{2A^2}{1+A}$.

VII. Results

A. Cross validation

In this experiment, we trained and tested the classifiers using 5-fold cross-validation as described above. The best set of parameters for each classifier were selected and the resulting AUC value (averaged over the 5-fold validation) is shown in Table III. Table III also shows standard deviations computed using cross-validation. For both single and windowed inputs, the C4.5-Rules have the best performance. In addition, due to the different size of the training sets (see Table I), almost all classifiers have smaller standard deviations with windowed inputs than single inputs.

TABLE III
CROSS VALIDATION WITH DIFFERENT CLASSIFIERS.

input	classifier	Mean of AUC	std
single	SLN	74.41	2.04
	SVM	78.36	1.8
	C4.5-Rules	86.55	1.21
windowed	SLN	75.94	0.59
	SVM	75.14	0.31
	C4.5-Rules	87.01	1.24

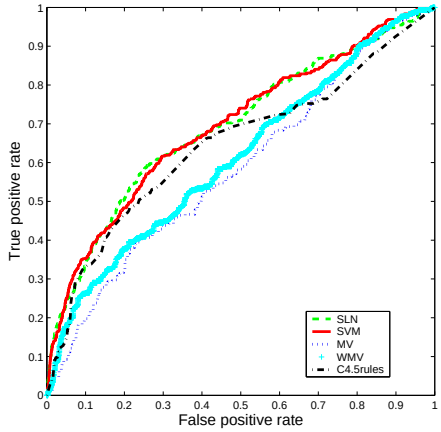

Fig. 3. ROC graph: five classifiers applied to the consistent test set with single inputs.

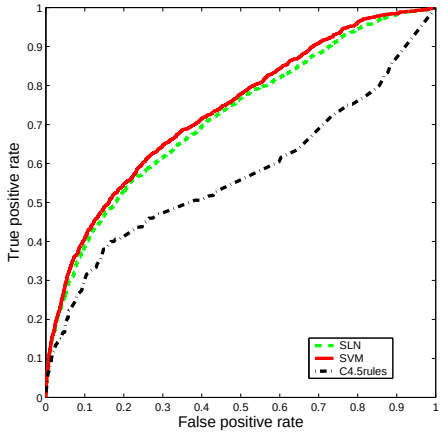

Fig. 4. ROC graph: three classifiers applied to the full test set using windowed inputs.

B. Classification results on the fixed consistent test set with single and windowed inputs

This test set has 5966 data points (see section III) in both the single and windowed versions.

The results are shown in Table IV, together with the best base algorithm (the one with the highest F-score). Compared with the best base algorithm, all classifiers, except MV and WMV decrease the fp_rate. It can be seen that with both single and windowed inputs, the SVM is clearly the best classifier - it outperforms the others in terms of the F-score and AUC performance metrics. However this is at a cost: in comparison to the best base algorithm the recall has been decreased, especially with single inputs. The classifier has become more conservative, predicting binding sites less often but with greater accuracy. When comparing the single and windowed results the only major difference is that C4.5-Rules does a lot better with single input data.

Figure 3 shows ROC curves obtained using the consistent test set and single inputs. The curves show that the SVM and SLN have similar performance, outperforming the others. MV and WMV are the weakest.

C. Classification results on the full test set with single and windowed inputs

In this experiment, we use the full contiguous test set. All the results are presented in Table V, with one of the corresponding ROC curves shown in Figures 4.

Looking at the results for the single inputs, the SLN performs well in the AUC, while the SVM in F-score. Although their recalls are lower than the best base algorithm, this is explained by their far lower fp_rate. With windowed inputs the story is very much the same. In fact the windowed SVM is the overall best performer across single and windowed classifiers. The C4.5-Rules perform particularly poorly, as is shown in Figure 4, where over some of the range it is predicting below random.

VIII. Conclusions

The significant result presented here is that by integrating the 12 algorithms we can considerably improve binding site prediction. In fact when considering the full contiguous test set, we are able to reduce the false positive predictions of the best base algorithm by 26%, whilst maintaining about the same number of true positive predictions. As expected the SVM gave a better classification result than the SLN and the decision trees. Majority voting was actually worse than the best individual algorithm. However, weighted majority voting was a little better. C4.5 has a tendency to badly overfit the training data and produce very poor predictions, sometimes worse than random.

Future work will investigate i) searching for a method to find out a suitable ratio of minority to majority classes, which could give better results; ii) using algorithm based technologies to cope with the imbalanced dataset; iii) considering a wider range of supervised meta-classifiers or ensemble learning algorithms. Another important avenue to explore will be to examine the biological significance of the results and we are currently working on using a visualisation tool.

References

[1] A. Apostolico, M. E, Bock, S. Lonardi and X. Xu, "Efficient Detection of Unusual Words," *Journal of Computational Biology*, Vol.7, No.1/2, 2000.

[2] M. I. Arnone and E. H. Davidson, "The hardwiring of development: Organization and function of genomic regulatory systems", Development 124, 1851-1864, 1997.

[3] T. L. Bailey and C. Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolymers," *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, 28-36, AAAI Press, 1994.

[4] C. M.Bishop. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press, New York.

TABLE IV

COMMON PERFORMANCE METRICS (%) TESTED ON THE SAME CONSISTENT POSSIBLE BINDING SITES WITH SINGLE AND WINDOWED INPUTS SEPARATELY.

input	Classifier	recall	precision	F-score	Accuracy	fp_rate	AUC±se
single	best Alg.	40.95	17.46	24.48	82.22	14.66	-
	MV	43.10	13.14	20.14	75.95	21.57	58.54±1.50
	WMV	41.19	17.35	24.42	82.05	14.86	61.28±1.51
	SLN	28.81	22.16	25.05	87.86	7.66	69.27±1.47
	SVM	32.14	24.46	**27.78**	88.23	7.52	69.49 ±1.47
	C4.5-Rules	29.29	23.08	25.81	88.15	7.39	64.59±1.50
windowed	SLN	34.29	18.87	24.34	85.00	11.16	69.21±1.47
	SVM	38.81	20/25	26.61	84.93	11.58	**72.58**±1.44
	C4.5-Rules	23.57	18.64	20.82	87.38	7.79	58.94±1.50

TABLE V

COMMON PERFORMANCE METRICS (%) TESTED ON THE FULL TEST SET WITH SINGLE AND WINDOWED INPUTS.

input	Classifier	recall	precision	F-score	Accuracy	fp_rate	AUC±se
single	best Alg.	36.36	18.40	24.44	85.97	10.73	-
	MV	35.73	15.12	21.25	83.48	13.35	61.66±0.81
	WMV	34.75	20.04	25.42	87.28	9.23	63.75±0.81
	SLN	25.19	25.09	25.14	90.64	5.01	70.54± 0.79
	SVM	27.91	26.97	27.43	90.79	5.03	69.30±0.79
	C4.5-Rules	23.03	23.14	23.08	90.43	5.09	60.45±0.81
windowed	SLN	31.82	22.66	26.47	88.97	7.23	71.93±0.78
	SVM	36.78	23.50	**28.67**	88.58	7.97	**73.65**±0.77
	C4.5-Rules	22.26	19.70	20.90	89.49	6.04	57.21±0.81

[5] M. Blanchette and M. Tompa, " FootPrinter: a program designed for phylogenetic footprinting," *Nucleic Acids Research*, Vol. 31, No. 13, 3840-3842, 2003.

[6] M. Buckland and F. Gey, "The relationship between Recall and Precision," *Journal of the American Society for Information Science*, Vol. 45, No. 1, pp. 12–19, 1994.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling Technique," *Journal of Artificial Intelligence Research*. Vol. 16, pp. 321-357, 2002.

[8] R. Fawcett, "ROC graphs: notes and practical considerations for researchers," Kluwer Academic publishers, 2004.

[9] T. Fawcett, "Using rule sets to maximize ROC performance," *Proceedings of the IEEE International Conference on Data Mining (ICDM-2001)*, Los Alamitos, CA, pp 131-138, IEEE Computer Society, 2001.

[10] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," *Radiology*, 143, 29-36, 1982.

[11] J. D. Hughes, P. W. Estep, S. Tavazoie and G. M. Church, "Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae," *Journal of Molecular Biology*, Mar 10;**296**(5):1205-1214, 2000.

[12] Wu, G and Chang, E. Y.: Class-boundary alignment for imbalanced dataset learning. *Workshop on learning from imbalanced datasets, II, ICML*, Washington DC, 2003.

[13] N. Japkowicz, "Class imbalances: Are we focusing on the right issure?" *Workshop on learning from imbalanced datasets, II, ICML*, Washington DC, 2003.

[14] M. Joshi, V. Kumar and R. Agarwal, "Evaluating Boosting algorithms to classify rare classes: Comparison and improvements," *First IEEE International Conference on Data Mining*, San Jose, CA, 2001.

[15] M. Markstein, A. Stathopoulos, V. Markstein, P. Markstein, N. Harafuji, D. Keys, B. Lee, P. Richardson, D. Rokshar and M. Levine, "Decoding Noncoding Regulatory DNAs in Metazoan Genomes", *proceeding of 1st IEEE Computer Society Bioinformatics Conference (CSB 2002)*, Stanford, CA, USA, 14-16, August, 2002.

[16] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kauffman, 1993.

[17] N. Rajewsky, M. Vergassola, U. Gaul and E. D. Siggia, "Computational detection of genomic cis regulatory modules, applied to body patterning in the early Drosophila embryo," BMC *Bioinformatics*, **3**:30, 2002.

[18] B. Scholköpf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, 2002.

[19] Y. Sun, M. Robinson, R. Adams, P. Kayes, A. G. Rust and N.Davey, "Integrating binding site predictions using meta classification methods", *Proceedings ICANNGA05*, 2005.

[20] G. Thijs, K. Marchal, M. Lescot, S. Rombauts, B. De Moor, RouzP and Y. Moreau, "A Gibbs Sampling method to detect over-represented motifs in upstream regions of coexpressed genes," *Proceedings Recomb'2001*, 305-312, 2001.

[21] T. F. Wu, C. J. Lin and R. C. Weng, "Probability Estimates for multi-class classification by pairwise coupling," *Journal of Machine Learning Research*, 5 pp. 975-1005, 2004.

[22] *http://www.hgmp.mrc.ac.uk/Software/EMBOSS/.*

[23] *http://family.caltech.edu/SeqComp/index.html.*