Using Randomised Vectors in Transcription Factor Binding Site Predictions

Faisal Rezwan, Yi Sun, Neil Davey , Rod Adams
School of Computer Science
University of Hertfordshire
College Lane, Hatfield
Hertfordshire AL10 9AB
{F.Rezwan, Y.2.Sun, N.Davey}@herts.ac.uk

Alistair G Rust
Wellcome Trust Sanger Institute,
Wellcome Trust Genome Campus,
Hinxton, Cambridge, CB10 1SA, UK
ar12@sanger.ac.uk

Mark Robinson
Department of Biochemistry and Molecular Biology,
Michigan State University,
East Lansing MI 48824, USA
blobby@msu.edu

***Abstract*— Finding the location of binding sites in DNA is a difficult problem. Although the location of some binding sites have been experimentally identified, other parts of the genome may or may not contain binding sites. This poses problems with negative data in a trainable classifier. Here we show that using randomized negative data gives a large boost in classifier performance when compared to the original labeled data.**

Keywords- Genes; Binding Site; Classification; Support Vector Machines

I. INTRODUCTION

Genes are regulated (turned on and off) according to whether specific sites on the genome have a regulatory protein attached to them, see Figure 1. These sites, called *binding sites* (or more technically: *transcription factor binding sites, TFBS*), are therefore critical in the way cells and their genes interact. Unfortunately locating the binding site(s) for a particular gene is difficult [1]. They maybe upstream or downstream of the gene and may be located some way from it. Moreover a specific regulatory protein may have many sites that it binds to in the genome, but these sites will not have a unique genetic signature (common DNA sequence). To experimentally find binding sites is costly and time consuming, so biologists use algorithms to predict where a binding site might be. For example if a short sequence of base pairs (*bp*) occurs in the genome of several different species then it is probably important and could be a binding site. In fact there are several different ways to predict the presence of a binding site [2, 3]. Here, and in earlier work [4], we combine the results of a group of different individual predictors to produce a prediction that is better than that of any of the individual predictions.

The major problem with training a classifier on these combined predictions is that the data can be rather unreliable. Whilst the labeling of a known binding site is normally correct (as it has been experimentally verified), the labeling of the other class may be much more dubious. This is the major issue we address here. We find that changing the negative training vectors can give a large benefit in the performance of the classifier on unseen data.

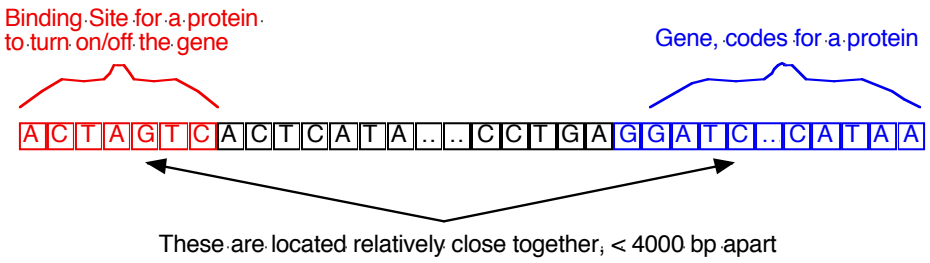

Figure 1. A gene and a binding site for a protein that regulates it.

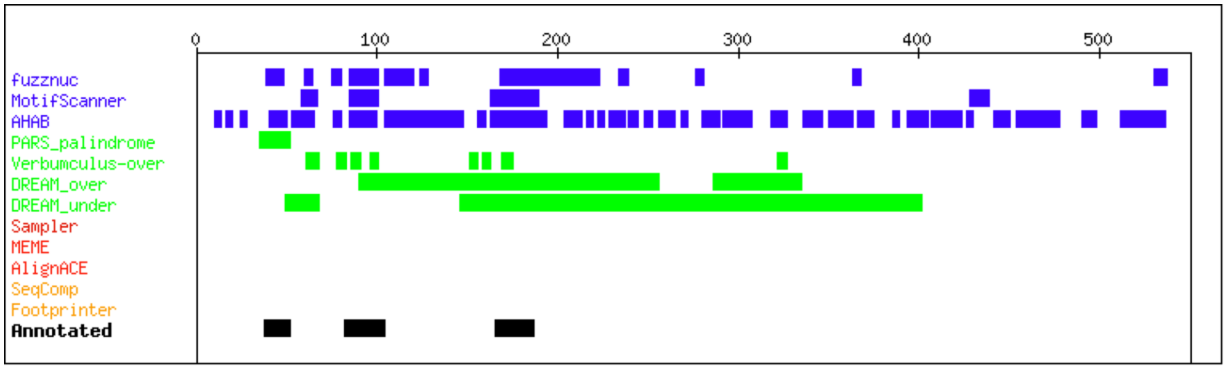

Figure 2. The predictions of the algorithms over a stretch of 550 base pairs in a yeast promoter region. The last row shows the actual position of the known binding sites

II. BACKGROUND

In this work we use data taken from the yeast genome and also from the mouse genome. These two species were chosen firstly because a lot is already known about their genomes and also to test our method on both a relatively simple regulatory regime, as found in yeast, and on a complex cellular mechanism as found in the mouse. We have discussed the data used in [5] so here just give a summary.

For each gene there is a corresponding *promoter* region where the binding sites will probably be located. The yeast data set consists of 112 of these regions, with average length 605 *bps*. Moreover 400 binding sites are known to exist in this data set. We also have the results of 12 different binding site prediction algorithms for each *bp* position. The algorithms predict the probability that a given *bp* is part of a binding site. Figure 2 shows the 12 algorithmic predictions together with the annotated binding sites. None of the individual algorithms are very accurate, but we hope that collectively they will do better than individually.

By simply concatenating the 12 prediction values we get a 12-ary prediction vector for each position and the label of each vector is the known status of that position: being in a binding site or not. This is illustrated in Figure 3.

The yeast data set is summarized in Table 1. As can be seen the proportion of locations that are part of a binding site is low, 7.8%. This is therefore imbalanced data[6], and we have reported elsewhere [7] how we use sampling to deal with this problem

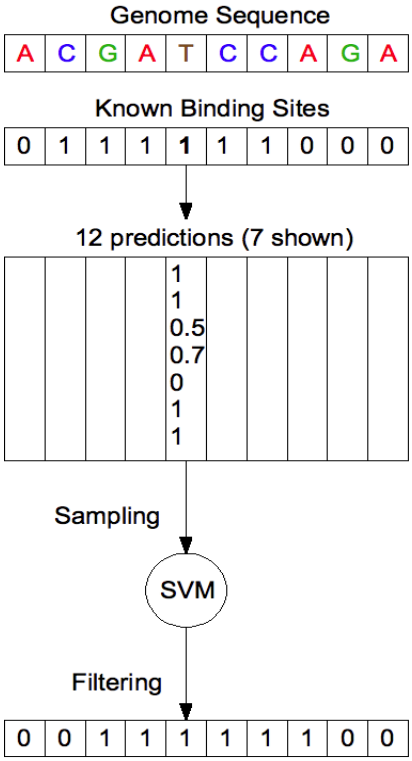

Figure 3. At each position in the yeast genome sequence we have an annotation and 12 algorithmic predictions of that annotation. We train an SVM to produce the annotation given the 12 algorithmic predictions.

TABLE I. A SUMMARY OF THE YEAST DATA

Total number of sequences	112
Total sequence length	67,851 bp
Average sequence length	605 bp
Average number of TFBS sites per sequences	3.6
Average TFBS width	13.2 bp
Total number of TFBS sites	400
Number of unique TFBS sites	69
TFBS density in total dataset	7.8%

The mouse data is summarized in Table 2. For this genome we have 7 algorithmic predictions and again we

concatenate them together to give a 7-ary prediction vector for each position. This data set is even more imbalanced than the yeast data – here only 2.85% of the data belong to a binding site.

TABLE II. A SUMMARY OF THE MOUSE DATA

Total number of sequences	47
Total sequence length	60,851 bp
Average sequence length	1295 bp
Average number of TFBS sites per sequence	2.87
Average TFBS width	12.78 bp
Total number of TFBS sites	135
TFBS density in total dataset	2.85%
Total number of sequences	47

A deeper problem with these data sets is that whilst one can be reasonably confident that the *bp*s labeled as being part of a binding site probably are, no such confidence can be extended to the rest of the promoter region. There may be many, as yet undiscovered, sites therein. This implies that the 0 labels could be incorrect on many vectors.

In both data sets there are a number of vectors that are repeated. The large amount of repeating data comes about because the original algorithms often produce the same prediction over long sequences of the genome, and so for regions where none of the algorithms change their prediction the training vectors will all be repeats. For example the all zero vector in the mouse dataset (in which all seven algorithms predict that the bp is not part of a binding site) occurs 6,356 and therefore makes up about 20% of the data. Moreover on 19 of these sites the biological annotation is that it is part of a binding site. We call vectors of this type (repeats that occur in both classes) *inconsistent* vectors. There are also repeats that occur in only one class and these we call simple *repeats*. The breakdown of the two data sets is given in Table 3. Removing the repeats from the consistent data gives the unique data points.

TABLE III. VECTORS IN THE TWO DATA SETS

Species	#Original	Inconsistent	Consistent	Unique
Yeast	67,851	46,695 (69%)	21,156 (31%)	6,521 (9.6%)
Mouse	60,851	12,119 (20%)	48,732 (80%)	32,747 (54%)

It can be seen that the yeast data set has very many inconsistent data points, and this suggests that this data set is particularly unreliable. In the work presented here we take the simplest approach to dealing with the inconsistent and repeated data – we remove all such vectors (whilst keeping one copy of the consistent vectors). This means that we lose over 90% of the Yeast data and 46% of the Mouse data. From a biological point of view this is unsatisfactory as a prediction is required for every base pair in the genome, but in this paper we are taking a machine learning perspective and want to evaluate our method on clean data. After cleaning the data the yeast data 885 (14%) of the 6,521 vectors are in the binding site class and for the mouse data the binding site class has 1,484 (4.5%) of the 32,750 vectors.

As we have already discussed the negative data may be particularly unreliable. So we investigated whether modifying the negative *training* data, so that it contained only vectors that were highly unlikely to be part of a binding site, could improve performance.

So we construct a synthetic *training* set of negative examples. It is important to note that the final *test* set is never altered in any way and therefore consists of unique vectors from the original data. We place all the training vectors labeled with a zero in a matrix. The matrix has one row for each prediction (7 for the mouse, 12 for yeast), and one column for each base pair. Each row is then independently randomly reordered. This effectively randomizes each vector whilst maintaining the overall statistical properties of each algorithm. For example if a particular algorithm produces a 0 prediction 95% of the time it will still do so after the randomization. It is presumably unlikely that a real binding site would elicit such randomly joined predictions. Doing this is not without its price. Although we can be reasonably confident that the vectors are now correctly labeled, all the information in the original non-binding site data has been completely lost. Our results will indicate how much information was actually present.

III. MEUSRING PEFORMANCE

As the data is imbalanced simple accuracy is not an adequate messure of performance. As is usual we take the confusion matrix and from it define the following measures.

	Predicted Negatives	**Predicted Positives**
Actual Negatives	True Negatives	False Positives
Actual Positives	False Negatives	True Positives

$$Recall = \frac{\text{True Positives}}{\text{Actual Positives}} \qquad Precision = \frac{\text{True Positives}}{\text{Predicted Positives}}$$

$$FP\_Rate = \frac{\text{False Positives}}{\text{Actual Negatives}} \qquad F\_Score = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

The *Recall* measures the proportion of the actual binding sites that are predicted, and the *Precision* measures the proportion of positive predictions that are correct. The *FP-Rate* measures the proportion of non-binding sites that are incorrectly predicted as being in a binding site. Biologists are keen to keep the *FP-Rate* low so as to avoid unecessary experimental testing. Finally the *F-Score* is a single number that rewards both high *Recall* and high *Precision*. Note that all four of these measures give values between 0 and 1, with in all cases except *FP-Rate* the higher value being better.

IV. THE CLASSIFIER

We use a standard Support Vector Machine (SVM) with a Gaussian kernel. As is well known such a classifier has two hyper parameters, the cost parameter, *C*, and *gamma* the width of the Gaussian kernel. These two parameters affect the shape and position of the decision boundary and it is important to find good values for a particular data set, and this is normally done by a process of cross-validation.

First a test set (1/3) is removed from the data. The remaining data is used for training/validation. This training set/test set spit is also validated with the whole process being repeated 3 times, once for each of the 3 partitions The remaining 2/3 is then partitioned, here into 5 folds, and a search is done for a pair of hyper parameters that performs well on each of the 5 folds. The full algorithm we use is as follows:

1. Remove repeats and inconsistent vectors
2. Split the data into 3 equal subsets, maintaining the relative frequency of each data class
3. For each of these 1/3 2/3 test set / training set splits:
    a. Split the training data into 5 partitions
    b. This gives 5 different training (4/5) and validation (1/5) sets.
    c. Randomise the negative training data, if required
    d. Use sampling to produce more balanced training sets.
    e. For each pair of *C/gamma* values F
        For each of the 5 training sets
            Train an SVM
            Measure performance on the corresponding validation set, exactly as the final test will be measured. So use the *F-Score*.
        Average the *F-Score* over the 5 trials
    f. Choose the *C/gamma* pair with the best average
    g. Reform the complete training set and train an SVM with the best *C/gamma* combination.
    h. Test the trained model on the unseen test set.

As described earlier in some of our experiments we replace non-binding site vectors, in the *training* set, with modified random vectors. We only do this in training sets: validation and test sets are never modified in any way. This is important: the validation sets are being used to find hyper-parameters that may give models that not only perform well on the validation sets but may also perform well on the actual test set.

V. THE EXPERIMENTS

A. *Exepriment 1 – The Original Yeast Data*

The original algorithms find it very difficult to predict binding sites correcty individually. The best single algorithm has an *F-Score* of 0.25 and the worst an *F-Score* of 0.04. Our first result, which repeats experiments previously published, shows that the combined predictor can give a small improvement in peformance, with an *F-Score* of 0.29.

TABLE IV. THE RESULT USING THE ORIGINAL YEAST DATA

Recall	Precision	FP-rate	F-score
0.77	0.19	0.57	0.29

Note that the high recall is achieved by over predicting the binding sites, hence the relatively low precision and high *FP-rate*.

B. *Experiment 2 –Yeast Data with Randomised Negative Data*

As stated earlier the negative data may not be accurately labeled. In this experiment the negative vectors in the training sets are replaced with randomized vectors, as described in the previous section. The results are shown in Table 5.

TABLE V. THE RESULT USING THE YEAST DATA WITH RANDOMISED NEGATIVE EXAMPLES

Recall	Precision	FP-rate	F-score
0.63	0.51	0.08	0.56

It can immediately seen that the change to the training set has had considerable impact on the classifier performance. The *F-Score* has been increased from 0.29 to 0.56. The new predictions have reasonable *Recall* and *Precision*. It is worth repeating at this point that the test set that the trained classifier is assessed on, has not been altered in any way, so that it has the same characteristics as the test set in Experiment 1. The trained classifiers are finding around 63% of the binding sites with incorrect predictions of less than 50% (*Precision* over 0.51). Importantly the predictor makes relatively few false positive predictions – so a biologist will not find themselves investigating many false predictions, with all the waste of time and resource that entails.

C. *Experiment 3 –Mouse Data*

The location of the binding sites on the mouse genome are also hard to predict for individual prediction algorithms. Using the seven algorithms together our best result is shown in Table 6. This result is not very good. The precision of the prediction is very low and the *FP-Rate* is too high. Many predicted binding sites in the test set are not so labelled.

TABLE VI. THE RESULT USING THE ORIGINAL MOUSE DATA

Recall	Precision	FP-rate	F-score
0.49	0.09	0.35	0.14

D. *Experiment 4 – Mouse Data with Randomised Negative Examples*

As stated earlier the negative data may not be accurately labeled. In this experiment the negative vectors in the training sets are replaced with randomized vectors, as described in the previous section. The results are shown in Table 7.

TABLE VII. THE RESULT USING THE ORIGINAL MOUSE DATA

Recall	Precision	FP-rate	F-score
0.77	0.68	0.03	0.69

We found the result of this experiment to be very surprising. The improvement in the performance of the classifier is even more pronounced here than on the yeast data. Perhaps most interestingly the *Precision* of the prediction shows very marked improvement. Unlike the yeast data this did not come at any cost to *Recall* which in fact also improves by roughly 50%. The relatively high *Precision* of the predictions then gives rise to a much reduced *FP-Rate* – just 3% of the predicted binding sites are in error.

VI. DISCUSSION

Our major result here is obviously the considerable affect of changing the training data so that the non-binding site class consists of randomly assigned algorithmic results. This was particularly pronounced with the mouse data. In fact the result was so dramatic that we have repeated the whole experiment several times to make sure that the result was reliable. How can this improvement in performance come about? It certainly implies that the original negative data is, in this case, actually unhelpful to the trainable classifier. However great care needs to be taken in the interpretation of the results. If it is the case that the negatively labeled vectors in the data (and therefore in the test set) are unreliable then any statistic using the first row of the confusion matrix is also unreliable. This means that the *Recall*, *Precision* and *F-Score* measures are probably reasonable, but also that the *FP-Rate* could be incorrect.

Of course our approach is similar to using a one class classifier, such as an SVDD classifier [8]. However our attempts to use such classifiers have not been anything like as successful as the approach we have described in this paper. The results obtained were similar to the original two class SVM.

The sites where we predict the presence of a binding site, which are not labeled as such, could be of interest to experimental biologists as potentially interesting parts of the genome. It would be interesting for us to find out if we were predicting the presence of previously unknown binding sites.

REFERENCES

1. Tompa, M., et al., *Assessing computational tools for the discovery of transcription factor binding sites.* Nat Biotechnol, 2005. **23**(1): p. 137-44.
2. Bailey, T.L. and C. Elkan, *Fitting a mixture model by expectation maximization to discover motifs in biopolymers.* Proc Int Conf Intell Syst Mol Biol, 1994. **2**: p. 28-36.
3. Blanchette, M. and M. Tompa, *FootPrinter: A program designed for phylogenetic footprinting.* Nucleic Acids Res, 2003. **31**(13): p. 3840-2.
4. Sun, Y., et al. *Using Real-Valued Meta-classifiers To Integrate Binding Site Predictions*. in *IJCNN*. 2005. Montreal, CA.
5. Robinson, M., et al., *Improving computational predictions of cis-regulatory binding sites.* Pac Symp Biocomput, 2006: p. 391-402.
6. Japkowicz, N. *Class imbalances: Are we focusing on the right isse?* in *Workshop on learning from imbalanced datasets, II, ICML*. 2003. Washington DC.
7. Sun, Y., et al. *Integrating binding site predictions using non-linear classification methods*. in *Machine Learning Workshop*. 2005. Sheffield: LNAI.
8. David, M.J.T. and P.W.D. Robert, *Support Vector Data Description.* Mach. Learn., 2004. **54**(1): p. 45-66.