Effect of Using Varying Negative Examples in Transcription Factor Binding Site Predictions in the Mouse Genome

Faisal Rezwan*[1] and Yi Sun[1] and Neil Davey[1] and Rod Adams[1] and Alistair G. Rust[2] and Mark Robinson[3]

[1]School of Computer Science, University of Hertfordshire, College Lane, Hatfield, Hertfordshire AL10 9AB, UK
[2]Wellcome Trust Sanger Institute,, Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SA, UK
[3]Benaroya Research Institute at Virginia Mason, Seattle WA 98101, USA

Email: Faisal Rezwan*- F.Rezwan@herts.ac.uk; Yi Sun - Y.2.Sun@herts.ac.uk; Neil Davey - N.Davey@herts.ac.uk; Rod Adams - R.G.Adams@herts.ac.uk; Alistair G. Rust - ar12@sanger.ac.uk; Mark Robinson- mrobinson@benaroyaresearch.org;

*Corresponding author

Abstract

Background: Identifying transcription factor binding sites (TFBSs) computationally is a hard problem as it produces many false predictions. Combining the predictions from existing predictors can improve the overall predictions by using classification methods like Support Vector Machines (SVMs). But conventional negative examples (that is, example which is the part of non-binding sites) in this type of problem are highly unreliable.

Results: In this study, we used different types of negative examples. One class of the negative examples was taken from far away from the promoter regions, where the occurrence of binding sites is very low, and another one was produced by randomisation. Thus we observed the effect of using different negative examples in predicting transcription factor binding sites in mouse. We devised a modified cross-validation technique for this type of biological problem. Using different negative examples with modified cross-validation technique improved the classifier performance and therefore the classifier could provide better predictions.

Conclusions: Our analysis addressed three distinct areas in the research of TFBS predictions: integrating multiple sources of evidences and using classification technique to improve TFBS predictions; an improved cross-validation method; and an investigation of the sources of negative examples. The major contribution of the analysis can be stated quite simply that for the mouse genome, the position of binding sites with high confidence could be predicted using our technique and the predictions are of much higher quality than the predictions of the original

base algorithms.

Background

Gene expression levels can be regulated *in vivo* by DNA-binding proteins called transcription factors (TFs) and genes are turned on and off according to whether specific sites on the genome have these regulatory proteins attached to them [1]. Transcription factors are themselves encoded by genes and the resulting regulatory interconnections form complex systems known as Genetic Regulatory Networks (GRNs) [2]. The stretches of DNA to which transcription factors bind in a sequence-specific manner are called transcription factor binding sites (TFBSs).

The location of TFBSs within a genome yields valuable information about the basic connectivity of a GRN, and as such is an important precursor to understanding the many biological systems that are underlain by GRNs [3–7]. There are many experimental and computational approaches for identifying regulatory sites. These experimental techniques are costly and often time-consuming and therefore not amenable to a genome-wide approach [8, 9]. Experimental approaches that are genome-wide, such as *ChIP-chip* and *ChIP-seq*, are themselves dependent on the availability of specific antibodies and still require additional verification [10]. Computational approaches for the prediction of TFBSs are therefore essential to complement and guide experimental exploration and verification [8, 10–16]. However, these approaches are typically prone to predicting many false predictions, limiting their utility significantly [8,16]. Figure 1 shows different computational predictions on the mouse data (described in [17]) yielding a lot of false predictions when compared with the real binding sites (annotated). These computational strategies are diverse and incorporate differing sources of biological information in the predictive process. They all have their own strengths and weaknesses. Therefore, there is good reason to consider that the set of binding sites predicted correctly by the individual sources of evidence are likely to form non-identical sets. If these predictions do really complement each other, then they can potentially provide more significant information when taken in combination. Therefore, combining their outputs may lead to better predictions. If one algorithm misses any binding site another algorithm may be able to capture that site. There are a number of approaches where the results from different algorithms have been combined together to give improved predictions [18–21]. In earlier works [17,22–24], results of a group of different predictors have been combined to produce a prediction that is better than that of any of the individual predictions.

We treated the problem as a classical binary classification problem where the examples were divided

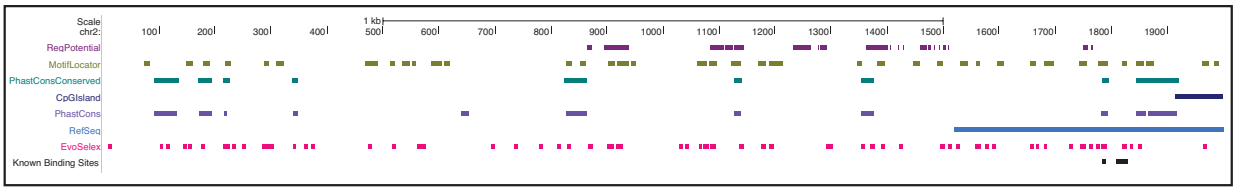

Figure 1: Visualisations of Predictions of the Seven Sources of Evidence on the Mouse Data. Here, 2000 bps upstream of the gene *Vim* has been shown. At each position in the genome sequence we have an annotation and seven algorithmic predictions of that annotation on the mouse data.

into two classes: the positive class contains those data that have been annotated as binding sites and the negative class otherwise. We ran experiments using the same positive examples but different sets of negative examples. The first set of negative examples is that part of the promoter regions that are not annotated as binding sites. But, in the type of biological data that are dealing with it is hard to establish an example as belonging to a negative class. So there might be a possibility of noise in the negative examples that we used in our first experiment, as base pairs marked as non-binding may be part of an unknown binding site and using them as negative examples might make the classifier perform incorrectly. To reduce this kind of noise we constructed two sets of negative examples- namely *distal* negative examples and *randomised* negative examples. This is the major issue we have addressed in this paper. We will also present the improvement of the performance of the classifier on both synthetic and unseen data by changing the negative vectors during training.

Method

Description of Data

Genomic Data

The mouse data set that we used has 47 annotated promoter sequences with an average length of 1294 *bps* (see Table 1). Most of the promoters are upstream of their associated genes and a few of them extend over the first exon including intronic regions. There are seven different prediction algorithms and biological evidence that had been used and combined together they made a single data set, a 60851 by 8 data matrix where the first column is the annotation label and rest of the columns represent scores from the seven prediction algorithms. The label is "1" if it is a part of a TFBS, otherwise "0" (see Figure 2). The algorithms and biological evidence are presented in Table 2 and discussed in one of our earlier works [22].

Table 1: A Summary of the Mouse Data. The TFBS density in the data clearly shows that the mouse data is imbalanced.

Total number of sequences	47
Total sequence length	60851 bps
Average sequence length	1294.70 bps
Average number of TFBSs per sequence	2.87
Average TFBSs width	12.78 bps
Total number of TFBS	135
TFBS density in total data set	2.85%

Strategy	Algorithms
Scanning Algorithms	MotifLocator EvoSelex
Evolutionary Algorithms	Regulatory Potential PhastCons (Conserved) PhastCons (Most Conserved)
Indirect Evidence	CpGIsland
Negative Evidence	Exon

Table 2: The seven sources of evidence used with the mouse dataset.

Problems with the Data and Solutions

From the statistics (in Table 3), it is quite clear that mouse data set is imbalanced as our feature of interest is significantly under-represented and it is likely to result in a classifier that has been over-trained on the majority class and can only act as a weak classifier for the minority class, which may give false classifications. To overcome this problem, we chose a databased method (described in [25]) in which the minority class (binding site examples) is over-sampled and majority class (non-binding site examples) is under-sampled. The Synthetic Minority Over-sampling Technique (SMOTE) [25] had been used to over-sample the minority class in the training set.

Table 3: Statistics of the Mouse Data. It shows that mouse data contains a lot of inconsistent and repeat vectors, which make the data unreliable for training.

Original	**Inconsistent**	**Repeat**	**Unique**
60,851	12,119(20%)	20,969 (34.5%)	32,747(54%)

Another problem with the data set is that whereas one can be reasonably confident that the *bps* labelled as being part of a binding site, 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 negative labels could be incorrect on

many vectors. In our data set there are also a number of vectors that are repeated. There are also repeats that belong to both classes, termed as *inconsistent* vectors, which make up about 20% of the data. There are also repeats that occur in only one class and these are simply called *repeats*. The vectors which occur exclusively once in any class and which do not belong to any repeats or inconsistent classes, are called *unique*. The breakdown of the mouse data set is given in Table 3. To deal with inconsistent and repeated data, we took the simplest approach by removing all such vectors (keeping one copy of the consistent vectors). As a result, we lost over 46% of the mouse data. From Figure 2 we can see that in the mouse dataset, one data vector is repeated 6,337 times in the negative example class labeled as the part of non-binding sites whereas the same vector is present 19 times in the positive example class where it is labeled as the part of binding sites. This single vector, which consists of all zeros, constitutes about 10.5% of the whole mouse dataset. There are also other inconsistent data vectors like this present in the dataset.

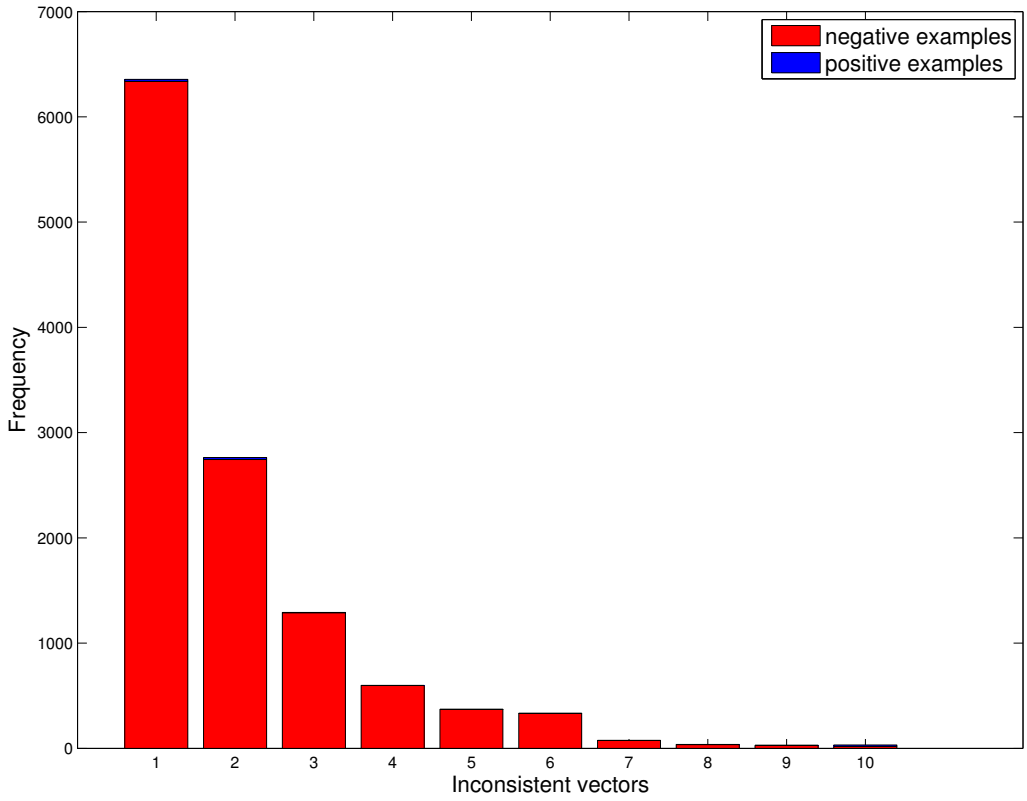

Figure 2: Frequency of inconsistent data rows in the mouse data. Only the inconsistent vectors with relatively higher frequency is shown here.

It is already been mentioned that negative data might be particularly unreliable. Therefore, we wondered whether we could improve the performance of the classifier by chaining the negative training data. We investigated whether modifying the negative training data set in such a way that it contained only vectors, that were highly unlikely to be part of a binding site, can improve performance. Hence we constructed two synthetic training sets with negative examples. It is important to note that the final test set has never been altered in any way and therefore consists of vectors from the original data. However, we deal with the test data in two different possible ways, which will be explained later.

For the first type of negative examples, called *distal negative examples*, we selected regions from the mouse genome that are at least 4500 to 5000 bps away from their associated genes. For the *randomised negative examples*, we placed all the training vectors labelled with a zero in the annotation into a matrix, as their probability of being part of a TFBSs is almost zero. Each column was then independently randomly reordered. This effectively randomised each vector whilst maintaining the overall statistical properties of each of our original prediction algorithms. It is unlikely that a real binding site would elicit such randomly joint predictions.

Combining Sources of Evidence

As mentioned in **Background** section , we have run three types of experiments:

Type 1: Using negative examples sequences not annotated as TFBSs from original data

Type 2: Replacing negative examples with distal negative examples

Type 3: Replacing negative examples with randomised negative examples

In each case, two sets of models were produced - one optimised for the filtered data and another one for the biological data. A detailed description of this has been given in the following section. In addition, we applied some pre-processing (data division, normalisation and sampling) on the training sets and some post-processing on the prediction sets. The whole process has been depicted in Figure 2.

Pre-processing: preparing training and test set

First we normalized values from each prediction and then searched for any repetitive or inconsistent data vector in the mouse data. Inconsistent vectors were eliminated from the training set, as these are unhelpful for the training process prediction results and likely to result in a classifier that has been trained on misleading support vectors. The repeats were also eliminated but keeping one representative instance. We also did the same after mixing the positive examples with the *distal* and *randomised negative examples*. After removing

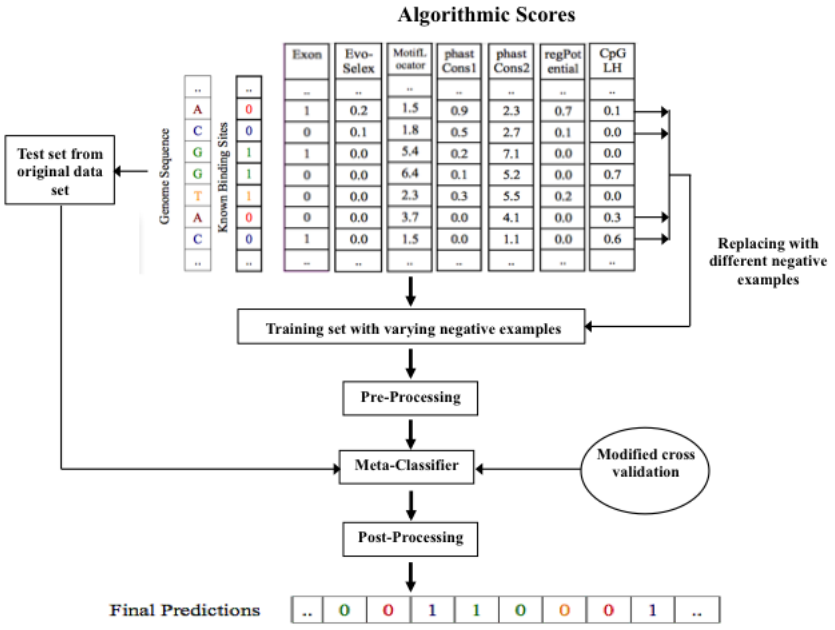

Figure 3: Complete Workflow of Combining the Sources of Evidence. We combined the scores from different algorithms and trained an SVM to produce the predictions.

the repetition and inconsistency from the data set, we randomised the data rows to mix the positive and negative examples. We used two-third of this new data set as training data. However, we kept one copy of the original dataset for the test dataset. We used sampling techniques on the training set to make it more balanced. This is done because if a dataset is imbalanced, then training is likely to result in a classifier that has been over-trained on the majority class and can only act as a weak classifier for the minority class [26,27]. In this sampling, the final ratio between the majority class to the minority class was was either $1:1$ or $1:2$ and this was done to explore the better margin of positive and negative classes suitable for training.

As mentioned earlier, while dealing with the test data, there are two different types of test sets we used. One is where only the consistent data points considered and any inconsistent and repetitions had been removed. This test set will be denoted as "filtered test set". The filtered test set will demonstrate the correct efficiency of the classifier. This test set will be of interest to machine learning practitioners, as it will demonstrate the classification efficiency of our SVM models on the data suitable for machine learning. The second test, denoted as "biological test set", is by considering data including repeats and inconsistent vectors t to give a biologically meaningful contextualised genome sequence. Whilst we realised that this data

set contain a lot of repetitions and inconsistent data vectors, it is realistic that the measures on this set will show how our process will work on real world data. Therefore, using biological test set will demonstrate how good our prediction model is trained to predict binding sites from biological data and ultimately it is the biologists that are most interested in the practical application of our method. Therefore, we undertook six experiments, for two test sets for each three types of experiments.

Post-processing

The original biological algorithms predict contiguous sets of *bps* as binding sites. However in this study, each *bp* is predicted independently of its neighbouring nucleotides. As a result, the classifier outputs many short predictions sometimes even with a length of only one or two *bps*. Therefore, we removed (replaced the positive prediction with a negative one) predictions with a length equal to or smaller than a threshold value, and measured the effect on the performance. In the present study, a range of threshold values (from 4 to 7) is used rather than a single one in order to assess the feasible threshold size.

The classifier and its performance measures

After constructing the training set using pre-processing, we trained a Support Vector Machines using LIB-SVM [28] on the training data. We used an SVM with a Gaussian kernel. As is well known such a classifier has two hyper parameters, the cost parameter, C, and the width of the Gaussian kernel, *gamma*. These two parameters affect the shape and position of the decision boundary. It is important to find good values of the parameters, and this is normally done by a process of cross-validation.

Table 4: Confusion Matrix. It is a way of reporting the performance of the classifier such as an SVM.

	Predicted Negatives	**Predicted Positives**
Actual Negatives	True Negatives(TN)	False Positives(FP)
Actual Positives	False Negatives(FN)	True Positives(TP)

$$Recall = \frac{TP}{TP + FN} \quad (1) \qquad Precision = \frac{TP}{TP + FP} \quad (2)$$

$$F\text{-}score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (3) \qquad FP\text{-}rate = \frac{FP}{FP + TN} \quad (4)$$

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad (5)$$

As we dealt with a strongly imbalanced data set (only 2.85% in mouse data is TFBSs), simply using the *Accuracy* (correct classification) as the performance measure is inappropriate, because then predicting everything as non-binding sites would give a very good *Accuracy* rate. Other measures are more suitable for this classification problem. Taking account of both *Recall* and *Precision* using the *F-score* should give us a good measure of classification performance since the *F-score* is actually harmonic average of *Recall* and *Precision*. In addition reducing the *FP-rate* should also be another major concern verifying a classifier's performance. The performance measures are described in Equation (1) to (5) and the confusion matrix in Table 4.

Cross-validation

In our previous works on this problem, performance had been measured on the validation data by simply using classification *Accuracy*. However this may not be the most effective method. The trained model will have to perform well on the test set and here *Accuracy* is not used. We therefore decided to investigate what could happen if we measured performance on the validation set exactly as we did on the test set. That is we measured the performance measures after the predictions had been filtered on length and with the repeated/inconsistent vectors placed back in the validation set. The step-by-step description of exactly what we have done is shown in Pseudocode 1.

Pseudocode 1 Finding the best hyper-parameters with modified cross-validation method.

```
Remove all inconsistent and repeated data points
Split the data into two-third training from the new data set, one-third test from the original data set
Split the training data into 5 partitions
This gives 5 different training (four-fifth) and validation (one-fifth) sets. The validation set is drawn from the related original data set
Use sampling to produce more balanced training sets
for each pair of C/gamma values do
    for each of the 5 training sets do
        Train an SVM
        Measure performance on the corresponding validation set, exactly as the final test will be measured. So use the Performance Measure, after the predictions on the validation set have been filtered
    end for
    Average the Performance Measure over the 5 trials
end for
Choose the C/gamma pair with the best average Performance Measure
Pre-process the complete training set and train an SVM with the best C/gamma combination
Test the trained model on the unseen test set
Post-processing the final prediction
```

Results and Discussion

In all experiments in this paper, we chose two different criteria for cross-validation: *Accuracy* and *F-score* and compare the performances. We explored a set of ratios (negative examples : positive examples) for under sampling the negative examples and in the results given below the ratio that has given the best classification performance has been used.

As mentioned before, we used two types of test sets for our results on the three experiments we referred. The first set of results were generated using filtered test set. In these experiments, we do not have any biological contextualisation, therefore we did not apply any post-processing on the predictions from them. For the second type of test set which includes repeats and inconsistent vectors, we used four variations of cross-validations. First as a baseline, we cross-validated in the normal way using *Accuracy* as the performance measure. We then changed these criteria so that the *F-Score* was used, in accord with how the model will be assessed on the final training set. Both these validation methods can be combined with the post-processing to filter out short predictions when assessing the performance of the model on a validation set. Therefore there are six experiments we undertook in this work.

Before presenting our experimental results, let us see how the base algorithms perform for identifying *cis*-binding sites on the test sets we used in all the experiments. We calculated the performance measures of the seven algorithms discussed in [22] and the comparison of the performance measures are shown in Figure 4. Among seven sources of evidence, we took results from the best algorithm ,*EvoSelex* and the confusion matrix is shown in Tables 4. Therefore, the Performance measures of *EvoSelex* is:

Table 5: Confusion matrix of the best base algorithm *EvoSelex* on the mouse data.

	Predictive Negatives	Predictive Positives
Actual Negatives	TN = 14985	FP = 3139
Actual Positives	FN = 511	TP = 273

Table 6: The results from the best base algorithms on the mouse data

Base Algorithms	**Recall**	**Precision**	**FP-score**	**FP-rate**
EvoSelex	0.35	0.08	0.13	0.17

Results from the best base algorithm shows that it wrongly classified a lot of sites as binding sites which actually are not. As a result, the *Recall* is very high as well as the *FP-rate*, but the *Precision* is very low. Now we will discuss the results from our experiments.

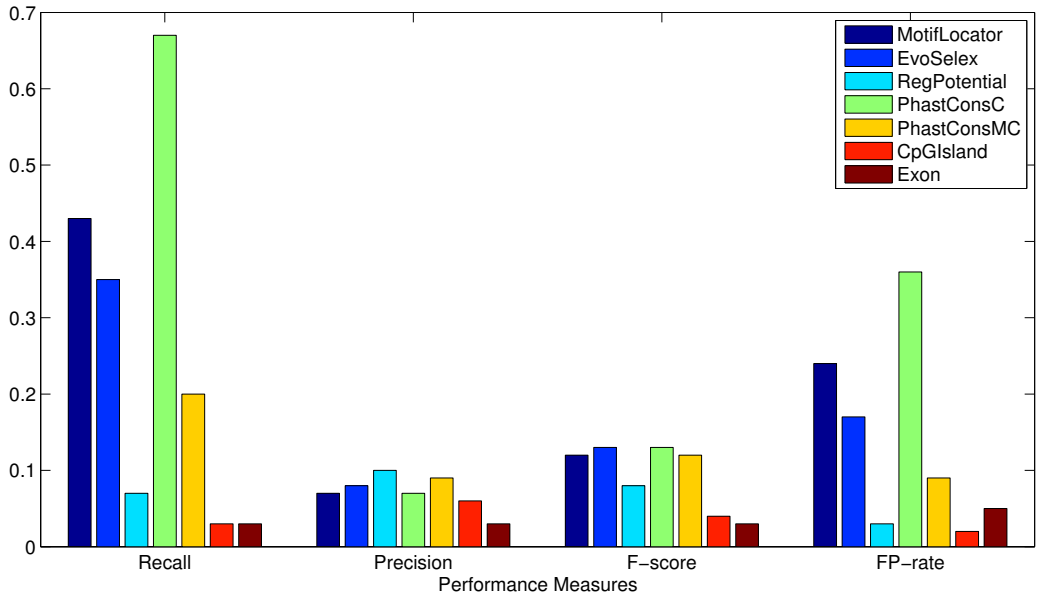

Figure 4: Comparison of performance measures of seven sources of evidence used on the mouse data. Among seven evidence, EvoSelex has the highest F-score and the lowest FP-rate.

Type 1 - Prediction using model from negative examples sequences not annotated as TFBSs from the original data

The results shown in Table 7 and Table 8 show a small improvement over the best base algorithm. For both data sets cross validating using the *F-Score* is a little better than using *Accuracy*.

Table 7: The result of using the original negative examples in the filtered test set with varying cross-validation methods

Cross-validation Criteria	Recall	Precision	F-score	FP-rate
Accuracy	0.30	0.09	0.14	0.14
F-score	0.44	0.11	0.18	0.15

Type 2- Prediction using model from promoter negative examples replaced by distal negative examples

Looking first at the results for the filtered data (Table 9) we can see that performance has improved and with the *F-Score* optimised model both *Recall* and *Precision* are much better giving a *F-Score* of 0.62. Even more remarkable is the model optimised for the full biological data (Table 10). In all cases the *Precision* is almost 1, implying that almost all the predicted binding sites are also annotated as such. The best models (all those except the first) also have quite a high recall and therefore have high *F-score*. The change to the

Table 8: The result of using the original negative examples in the biological test set with varying cross-validation methods

Cross-validation Criteria	Recall	Precision	F-score	FP-rate
Accuracy	0.20	0.14	0.16	0.05
F-score	0.24	0.17	0.20	0.04
Accuracy+post-processing	0.17	0.15	0.16	0.04
F-score+post-processing	0.24	0.17	0.20	0.05

negative training data can therefore be seen to be of significant benefit.

Table 9: The result of using the distal negative examples in the mouse data with varying cross-validation methods

Cross-validation Criteria	Recall	Precision	F-score	FP-rate
Accuracy	0.27	0.56	0.36	0.02
F-score	0.62	0.62	0.62	0.04

Table 10: The result of using the distal negative examples in the mouse data with varying cross-validation methods

Cross-validation Criteria	Recall	Precision	F-score	FP-rate
Accuracy	0.37	0.97	0.53	0.0006
F-score	0.65	0.99	0.79	0.0002
Accuracy+post-processing	0.68	0.99	0.81	0.0001
F-score+post-processing	0.68	0.99	0.81	0.0001

Type 3 - Prediction using model from promoter negative examples replaced by randomised negative examples

Surprisingly the randomised negative training data actually produces the best models for both types of data. With the filtered data (Table 11) we get a best *F-Score* of 0.82, reflecting a predictor with both high precision and recall. We also get a predictor of similar quality on the full test set (Table 13).

Table 11: The result of using the randomised negative examples in the mouse data with varying cross-validation methods

Cross-validation Criteria	Recall	Precision	F-score	FP-rate
Accuracy	0.80	0.79	0.79	0.02
F-score	0.82	0.83	0.82	0.02

Here is the confusion matrix of our best model on the filtered data:

Table 12: Confusion matrix of prediction using best model on the filtered test data.

	Predictive Negatives	Predictive Positives
Actual Negatives	TN = 10831	FP = 176
Actual Positives	FN = 191	TP = 836

Whereas, the result using biological test set is shown in Table 13.

Table 13: The result of using the randomized negative examples in the mouse data with varying cross-validation methods

Cross-validation Criteria	**Recall**	**Precision**	**F-score**	**FP-rate**
Accuracy	0.76	0.69	0.73	0.02
F-score	0.69	0.97	0.81	0.001
Accuracy+post-processing	0.76	1.0	0.86	0.00
F-score+post-processing	0.76	1.0	0.86	0.00

The confusion matrix of our best model on the biological data is as follows:

Table 14: Confusion matrix of prediction using best model on the biological test data.

	Predictive Negatives	Predictive Positives
Actual Negatives	TN = 18124	FP = 0
Actual Positives	FN = 190	TP = 594

It is interesting to see that the number of False Positives decreased to zero in case of biological test set whereas we can still observe some False Positives for filtered test set. This is because model on filtered test set predicts a lot of binding sites of very short in length. Whereas, in case of prediction on biological test set, we undertook the post-processing which eliminated all the short predictions. Therefore, the number of False Positives reduced to zero and produced 100% correctly predicted positive examples.

Comparison of the prediction performances

The first way of reporting, while using filtered test set, helps us to understand how good the classifier is after training it with new negative examples along with the modified cross-validation method. In this case, we are not going to compare the results with the best base algorithms as the base algorithms need to have repetitions and inconsistency in the datasets. The comparison (Figure 3) shows remarkable improvement on the classifier's performance. While using distal negative examples in the mouse data the *F-score* improves from 18% to almost 62%. This is due to the fact that the classifier improved almost all the measures (TP,

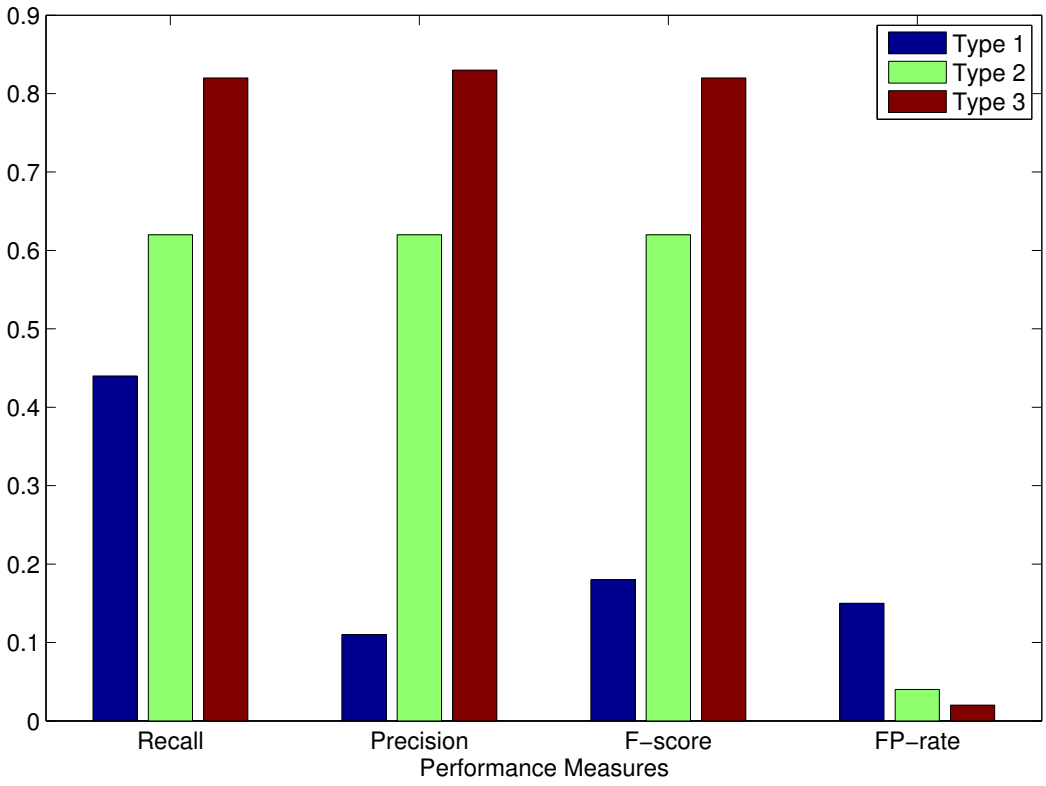

Figure 5: Comparison of prediction results on the filtered test set.

FP, and FN except TN) in the confusion matrix. The True Negative prediction is decreased by 50% as the previous classifier tried to predict everything as negative examples. However, the False Positive prediction increases three times which increases the proportion of correct predicted positive examples. The same is true when using random negative examples in the mouse data.

By extending the same approach while reporting classification performance on test sets with repeats and inconsistent vectors, we actually observed the same trend of improvements. Due to the nature of the data set (containing a lot of repeats), the prediction performances are more improved than that of with the test sets where we excluded repeats and inconsistent vectors. Accumulating all the results together (see Figure 5), we can observe that a more than three times improvement occurred in *F-score* by using randomised negative examples comparing to the best base algorithms. This is due to a huge drop in false predictions (FNs and FPs). The False Negative prediction is decreased by four times and there is very little False Positive prediction, in the case of mouse data it is even zero. Therefore, there is a big reduction in FP-rate.

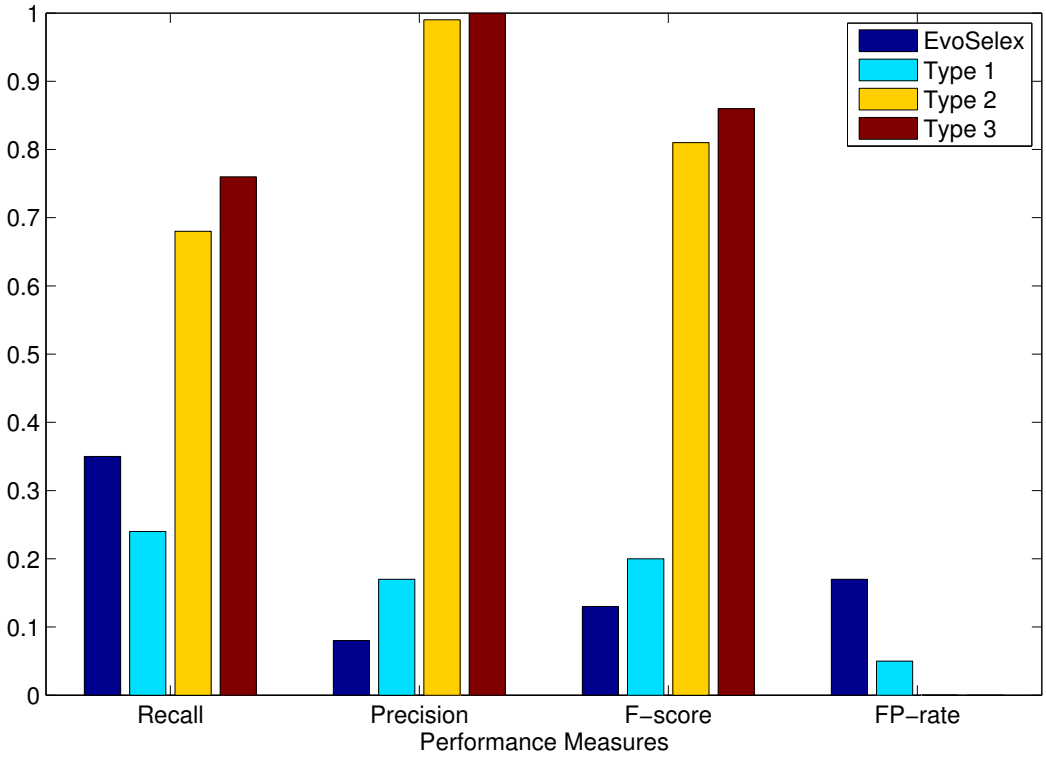

Figure 6: Comparison of prediction results on the biological test set.

This means that the possibility of predicting new novel sites become less. This can be due to the fact that the algorithms combined can characterise the annotated examples (examples from the positive class), but cannot characterise promoter negative examples well. These negative examples act as noisy data for the classifier, which is an obstacle to the classifier performance. However, this combination technique works fine with other negative examples (distal and randomised), which have less possibilities of having binding sites in them.

Visualisation of predictions on biological test sets

Apart from assessing the prediction based on performance measures, we visualised the data (like Figure 1) to see if our prediction was as good as it reflected in our results. We took a fraction of mouse genome (upstream region of the gene *MyoD1*, *Q8CFN5*, *Vim*, and *U36283*) and compared our best results from different experiments along with prediction algorithms and annotation. In Figure 6, the upper 7 results are from the original prediction algorithms and the next one is experimentally annotated binding sites from

ABS [29] or OregAnno [30]. The last three results are our best prediction results from three different types of experiments (described in Section 4). The figure shows that the prediction algorithms generate a lot of false predictions. On the other hand, using original mouse data (Experiment 1) does not make good predictions. Whereas, using *distal* or *randomised negative examples* (Experiment 2 or 3) improves the prediction considerably. The predictions are almost identical to annotations and experiment with *randomised negative example* gives slightly better predictions than that of *distal negative examples*.

Conclusions

The identification of binding sites for transcription factors in a sequence of DNA is a very difficult problem. In previous studies, it was shown that basic algorithms individually could not produce accurate predictions and consequently produced many false positives. Though the combination of these algorithms using two class SVM (see Experiment 1) gave better results than each individual prediction algorithm, there were still a lot of false positives due to vulnerability of the negative examples in our data set. Our current approach is similar to using a one class classifier. However our previous attempts to use such classifiers had results similar to that of the original two class SVM (see Experiment 1). Our present results show that a change in the provenance of the negative examples significantly improves the resulting predictions and that implementation of the new cross-validation technique can bring further improvements. Consequently our major result here is obviously the effects of changing the source of the negative examples used in the training data. Along with the novel cross-validation method our procedure can be a step in the right direction for dealing with this type of biological data. At present we are still in a preliminary stage of our work and any claim that we have solved the binding site problem would be premature, but the results so far certainly look promising. Future work will involve using the most recent prediction algorithms available and we would also like to extend our work to the genomes of other organisms.

Author's contributions

FR, YS, ND, RA, AR, and MR

Acknowledgements

Acknowledgements if any ...

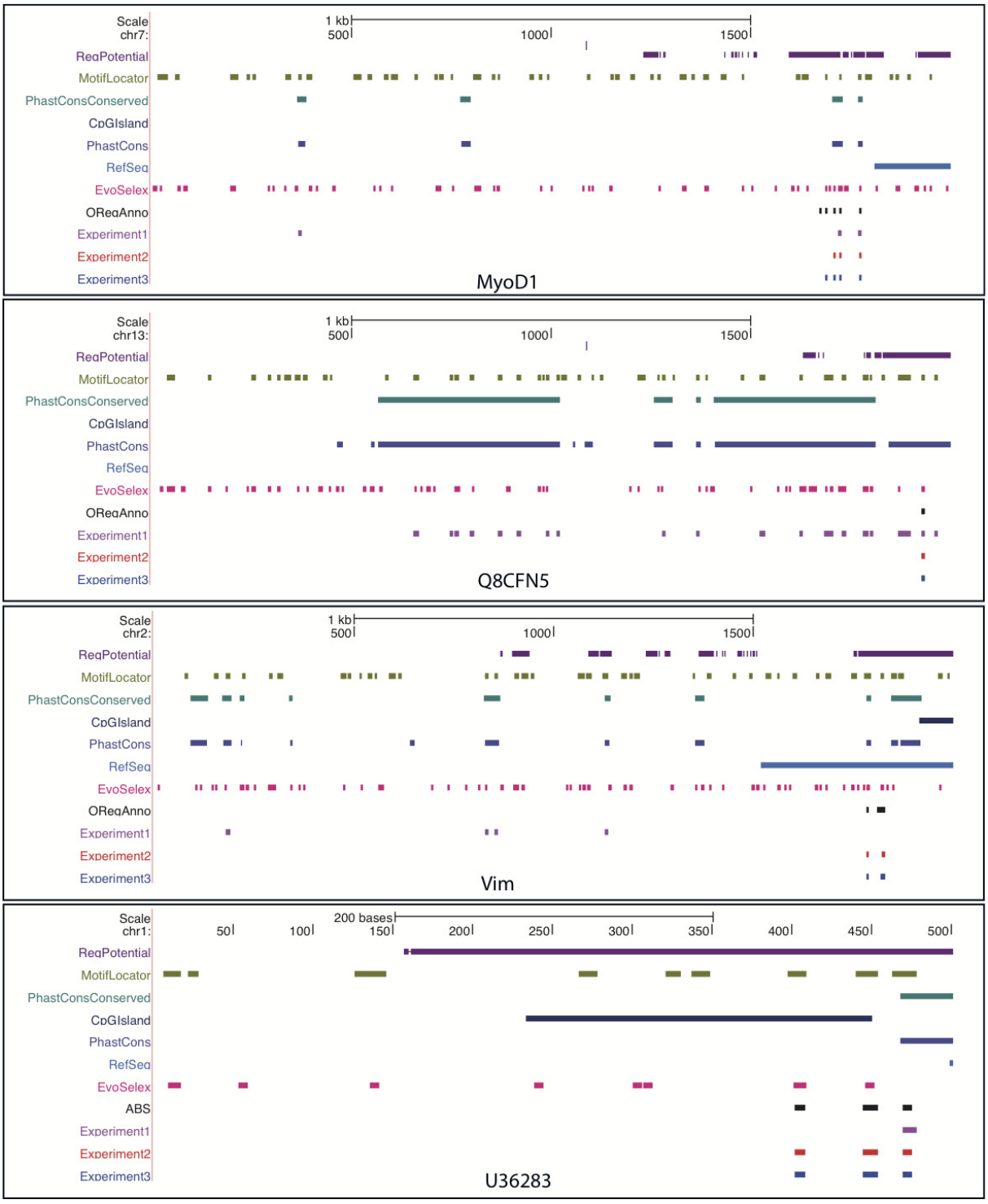

Figure 7: Visualisations of Predictions of the Seven Sources of Evidence and Varying Negative Examples with Modified Cross-validation Method on the Mouse Data. At each position in the genome sequence we have an annotation and several algorithmic predictions of that annotation.

References

1. White RJ: *Gene Transcription: Mechanism and Control*. Oxford, UK: Willey-Blackwell 2000.
2. Alberts B, Bray D, Lewis J, Raff M, Roberts K, Watson JD: *Molecular Biology of the Cell, 3rd edition*. New York: Garland Science 1994.
3. Arnone MI, Davidson EH: **The hardwiring of development: organization and function of genomic regulatory systems**. *Development* 1997, **124**:1851–1864.
4. Davidson EH: *Genomic Regulatory Systems: Development and Evolution*. Academic Press 2001.
5. Yuh CH, Bolouri H, Davidson EH: **Genomic cis-regulatory logic: experimental and computational analysis of a sea urchin gene**. *Science* 1998, **279**:1896–1902.
6. Ptashne M, Gann A: *Genes and Signals*. New York: Cold Spring Harbour Laboratory Press 2002.
7. Davidson EH: **A view from the genome: spatial control of transcription in sea urchin development**. *Curr. Opin. Genet. Dev.* 1999, **9**:530–541.
8. Tompa M, Li N, Bailey TL, Church GM, De Moor B, Eskin E, Favorov AV, Frith MC, Fu Y, Kent WJ, Makeev VJ, Mironov AA, Noble WS, Pavesi G, Pesole G, Regnier M, Simonis N, Sinha S, Thijs G, van Helden J, Vandenbogaert M, Weng Z, Workman C, Ye C, Zhu Z: **Assessing computational tools for the discovery of transcription factor binding sites**. *Nat. Biotechnol.* 2005, **23**:137–144.
9. Brown CT, Rust AG, Clarke PJ, Pan Z, Schilstra MJ, De Buysscher T, Griffin G, Wold BJ, Cameron RA, Davidson EH, Bolouri H: **New computational approaches for analysis of cis-regulatory networks**. *Dev. Biol.* 2002, **246**:86–102.
10. Elnitski L, Jin VX, Farnham PJ, Jones SJ: **Locating mammalian transcription factor binding sites: a survey of computational and experimental techniques**. *Genome Res.* 2006, **16**:1455–1464.
11. Wei W, Yu XD: **Comparative analysis of regulatory motif discovery tools for transcription factor binding sites**. *Genomics Proteomics Bioinformatics* 2007, **5**:131–142.
12. Nguyen TT, Androulakis IP: **Recent Advances in the Computational Discovery of Transcription Factor Binding Sites**. *Algorithms* 2009, **2(1)**:582–605.
13. Das MK, Dai HK: **A survey of DNA motif finding algorithms**. *BMC Bioinformatics* 2007, **8 Suppl 7**:S21.
14. Pavesi G, Mauri G, Pesole G: **In silico representation and discovery of transcription factor binding sites**. *Brief. Bioinformatics* 2004, **5**:217–236.
15. Blanchette M, Schwikowski B, Tompa M: **Algorithms for phylogenetic footprinting**. *J. Comput. Biol.* 2002, **9**:211–223.
16. Hu J, Li B, Kihara D: **Limitations and potentials of current motif discovery algorithms**. *Nucleic Acids Res.* 2005, **33**:4899–4913.
17. Sun Y, Robinson M, Adams R, te Boekhorst R, Rust AG, Davey N: **Integrating genomic binding site predictions using real-valued meta classifiers**. *Neural Comput. Appl.* 2009, **18**:577–590, [http://portal.acm.org/citation.cfm?id=1666489.1666493].
18. Che D, Jensen S, Cai L, Liu JS: **BEST: binding-site estimation suite of tools**. *Bioinformatics* 2005, **21**:2909–2911.
19. Huber BR, Bulyk ML: **Meta-analysis discovery of tissue-specific DNA sequence motifs from mammalian gene expression data**. *BMC Bioinformatics* 2006, **7**:229.
20. Romer KA, Kayombya GR, Fraenkel E: **WebMOTIFS: automated discovery, filtering and scoring of DNA sequence motifs using multiple programs and Bayesian approaches**. *Nucleic Acids Res.* 2007, **35**:W217–220.
21. Wilczynski B, Darzynkiewicz M, Tiuryn J: **MEMOFinder: combining de novo motif prediction methods with a database of known motifs**. *Nature Precedings* 2008, :1–6.
22. Sun Y, Robinson M, Adams R, Rust AG, Davey N: **Prediction of Binding Sites in the Mouse Genome Using Support Vector Machines**. In *ICANN (2)*, *Volume 5164 of* Lecture Notes in Computer Science. Edited by Kurková V, Neruda R, Koutník J, Springer 2008:91–100.

23. Sun Y, Castellano CG, Robinson M, Adams R, Rust AG, Davey N: **Using pre & post-processing methods to improve binding site predictions**. *Pattern Recogn.* 2009, **42**:1949–1958, [http://portal.acm.org/citation.cfm?id=1542560.1542848].

24. Robinson M, Castellano CG, Rezwan F, Adams R, Davey N, Rust A, Sun Y: **Combining experts in order to identify binding sites in yeast and mouse genomic data**. *Neural Networks* 2008, **21**(6):856–861.

25. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP: **SMOTE: Synthetic Minority Over-sampling Technique**. *J. Artif. Intell. Res. (JAIR)* 2002, **16**:321–357.

26. Chawla NV, Lazarevic A, Hall LO, Bowyer KW: **SMOTEBoost: Improving Prediction of the Minority Class in Boosting**. In *PKDD*, *Volume 2838 of* Lecture Notes in Computer Science. Edited by Lavrac N, Gamberger D, Blockeel H, Todorovski L, Springer 2003:107–119.

27. Japkowicz N: **Class imbalances: are we focusing on the right issue**. *Workshop on Learning from Imbalanced Data Sets II* 2003, :17–23.

28. Chang CC, Lin CJ: **LIBSVM: A library for support vector machines**. *ACM Transactions on Intelligent Systems and Technology* 2011, **2**:27:1–27:27. [Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm].

29. Blanco E, Farre D, Alba MM, Messeguer X, Guigo R: **ABS: a database of Annotated regulatory Binding Sites from orthologous promoters**. *Nucleic Acids Res.* 2006, **34**:D63–67.

30. Montgomery SB, Griffith OL, Sleumer MC, Bergman CM, Bilenky M, Pleasance ED, Prychyna Y, Zhang X, Jones SJ: **ORegAnno: an open access database and curation system for literature-derived promoters, transcription factor binding sites and regulatory variation**. *Bioinformatics* 2006, **22**:637–640.