A Simple Spatially Descriptive Feature Extraction Scheme For DNA Regulatory Sequences

Alessandro Luca Asoni
aasoni1@jhu.edu

16 December 2012

Abstract

This project introduces a simple scheme for feature extraction from mammalian enhancer DNA sequences that achieves a basic spatial or clustering description of the DNA elements. Using SVM it is established that introducing these features improves prediction accuracy from approximately 75% to 80%, compared to only using k-mer counts, suggesting that some new information may be captured by this scheme. The focus of this project was primarily to select predictive features among the new set, in order to find correlation to possible biological function. The most predictive features showed matches with known transcription factors, and one feature selection scheme resulted in an SVM test accuracy as high as 75%, even when dropping from over 100,000 to just 100 features.

1 Introduction

DNA sequences of length k, commonly known as k-mers, are often used as features for classification because they constitute a complete and unbiased feature set (Lee et al. 2011). However, a fundamental limitation of this technique is that as the length of these oligomers is increased, in order to capture spatial correlations in DNA sequence, the frequency of observing any specific k-mer becomes very small, leading to increasingly sparse feature vectors with binary counts. This leads to sensitivity of machine learning techniques to noisy estimation, when they are applied for classification.

In this project an alternative method is proposed, where features are extracted from raw DNA sequences associated with an enhancer signal in a way that hopes to capture not only local characteristics, but also a spatial description of enhancer sequence.

1.1 Enhancer Description

Enhancers are DNA regions that have a regulatory function on gene transcription. The interaction between enhancers and genes is often complex, and this mode of interaction is well understood only for a limited subset of enhancer-gene couples (Benerji 1981). In eukaryotic transcription control, small proteins known as transcription-factors (TFs) bind short DNA sequences (4 to 6 base pairs) within an enhancer region. TFs can then recruit larger protein complexes that interact with sections of DNA that may be at an arbitrary distance away, initiating transcription of genes at these sites. Transcription factor binding sites have been used successfully for DNA region classification (Matys et al. 2003), but because of the variety in the mode of binding of larger protein complexes to TFs, and the importance of their spatial placement on DNA, using k-mers has shown better results (Lee at al. 2011).

1.2 Why Is Understanding Enhancer Dynamics Useful?

Genomic research has in recent years produced large data sets (1000 Genome Project) that have uncovered statistically significant links between a variety of diseases and specific DNA polymorphisms. These variations occur in intergenic sequences and are often suspected to alter the behavior of gene regulation by changing enhancer binding sites. Being able to understand the structure and functioning mode of these regulation complexes can give insight on the way diseases with high social impact (e.g. cancer, Alzheimer's and Parkinson's) develop and on new ways to treat them. In this project we present a set of predictive features that may indicate important structural characteristics of these enhancer complexes.

2 Data

The data that was used for this project consisted in several thousands mouse embryos EP300-bound DNA elements (Visel at al. 2009). These were collected using ChIP-seq technology, and consist in DNA sequences, approximately between 100 and 400 base pairs in length, that were determined to bind EP300, a protein that regulates gene activity in mammals. To create a negative set, several thousand DNA sequences between 100 and 400 bp were sampled using the UCSC Genome Browser, with care in trying to avoid regions with a high DNase hypersensitivity signal (another technique used for finding enhancers).

3 Classification

As will be explained below, feature extraction for this project resulted in a very rich feature set. So, in order to avoid the "curse of dimensionality", support vector machines (SVMs) were chosen as classification tool. The training time for an SVM is proportional to the number of examples and not the number of features, which makes it a better classifier for this problem then other algorithms. Another benefit of SVMs is that their optimization objective is convex, meaning that it's guaranteed to find the global optimum solution.

An SVM finds a decision boundary that separates the positive and negative training data. This boundary is a hyperplane that maximizes its distance from the elements of the two sets, known as the margin. Given N labelled vectors $\{\mathbf{x}_i, y_i\}$, $i \in \{1...N\}$, where $\mathbf{x}_i \in \mathcal{R}^n$ are feature vectors and $y_i \in \{+1, -1\}$ are labels, the decision boundary is found by minimizing $\|\mathbf{w}\|^2$ such that $y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1$. The optimal solution in practice is found by maximizing the dual form:

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i \alpha_i y_j \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

over α_i with constraints, $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$ (Joachims 1999). The prediction rule for an SVM in the dual form is then:

$$\hat{y} = \text{sign} \sum_{i=1}^{N} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x})$$

Constrained optimization on the dual leads to $\alpha_i = 0$ for correctly classified training examples, hence the above sum only needs to be computed over the so called *support vectors*, which correspond to the data points that lie on the maximum margin hyperplanes in feature space (Bishop 2006).

The focus of this project is not directly to achieve high classification accuracy. So the natural following step of parameter tuning (e.g. slack variables) or performance comparison with other training algorithms is left for future work. What was needed was a Machine Learning classification algorithm that was suited for the type of data collected, in order to evaluate if adding new dimensions to our data could improve the separability of the positive and negative training sets, indicating feature relevance.
The SVM implementation used was SVM light (Joachims, 1999).

4 Features

4.1 Feature Extraction

Because TFs are often between 4 and 6 base pairs in length, a basic k-mer of length 5 was used for feature extraction. The first set of features that was attributed to each enhancer and each negative sequence was a count of all possible 5-mers appearing in the corresponding DNA element (analogously as done by Lee at al.). The next step was then to find a new set of 'spatial' features that showed an improvement in classification. The end goal being to find a small subset of highly predictive spatially descriptive features that could be used to make some deductions about enhancer behavior and to improve classificaton accuracy.

This was done by choosing a second set of features that corresponded to the count of any two 5-mers appearing together in a window of a specified base pair length. By evaluating performance using SVM we found that the best improvement was achieved when the length of the window was set to 100 bp. This led to an increase in classification accuracy from 75.23% (precision/recall: 88.28%/66.43%) to 80.55% (precision/recall: 91.80%/73.26%) on test data. It also increased the number of features from a few hundred to over a 100,000, slowing down the classification process considerably.

The above improvement of 5% in accuracy seems to indicate that we have gained information by introducing the new set of features. Nonetheless these results do not inspire per se, because the increased accuracy comes at a high performance cost and we have yet to establish if this improvement can be attributed to a small subset of relevant features among the new large set that was introduced.

4.2 Feature Selection

4.2.1 Conditional Mutual Information Maximization

One of the more commonly used tools to assess which features are the most predictive relies on information theory, and is called *Mutual Information Maximization*. The mutual information (MI) between X and Y is defined as follows:

$$I(X,Y) = -\sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

which quantifies how much information is shared between X and Y. This formula is derived from the most fundamental concept in information theory, the entropy $H(X)$:

$$H(X) = -\sum_{x \in X} p(x) \log p(x)$$

Given the above definition, a small subset of k features that carry as much information as possible, could be found by maximizing the joint mutual information (JMI) for k features and the label Y:

$$\underset{X_1, X_2, ..., X_k}{\text{argmax}} \quad I(X_1, X_2, ..., X_k, Y)$$

However we face again the curse of dimensionality: if all X_i's are discrete and take η values and Y is binary, then the vector: $(X_1, X_2, ..., X_k, Y)$ has $2 \cdot \eta^k$ possible states. Hence an alternative method is required.

It can be shown that:

$$\begin{aligned} I(X_1, ..., X_k, Y) = I(X_1, ..., X_{k-1}, Y) \\ + I(X_k, Y | X_1, ..., X_{k-1}) \end{aligned}$$

This uses the concept of Conditional Mutual Information (CMI): $I(X_k, Y | X_1, ..., X_{k-1})$, which quantifies the shared information between X_k and Y given features $X_1, ..., X_{k-1}$. If we assume that we currently have already picked $k - 1$ features then, to maximize the JMI, we can pick the next feature to be the one that maximizes the CMI. So we can start by picking X_1 to be the feature that maximizes the MI $I(X_1, Y)$ and then can sequentially pick the next $k - 1$ features according to this scheme. Unfortunately calculating CMI, like MI, quickly becomes intractable as the dimensionality increases. Several algorithms exist that try to overcome this issue. For this project one was implemented (Wang at al. 2004) that uses the fact that the CMI between Y and some feature X^* given some other k features is always smaller than their mutual information given a smaller set of features:

$$I(X^*, Y | X_1, ..., X_k) \leq I(X^*, Y | \underbrace{X_i, ..., X_j}_{k-1})$$

Therefore $I(X^*, Y | X_1, ..., X_k)$ is estimated by their minimum value:

$$I(X^*, Y | X_1, ..., X_k) \approx \min I(X^*, Y | \underbrace{X_i, ..., X_j}_{k-1})$$

which can be further simplified using what is called a "triplet" in the following way:

$$\min I(X^*, Y | \underbrace{X_i, ..., X_j}_{k-1}) \approx \min_i I(X^*, Y | X_i)$$

The $k - 1$ features minimizing the CMI above are the ones most correlated with X^* meaning that, in order to maximize the predictive power of X^*, it has to be chosen such that $\min I(X^*, Y | \underbrace{X_i, ..., X_j}_{k-1})$ is as large as possible. So the final rule to pick a new feature X^* would be:

$$X^* = \underset{X^* \in \mathbf{X}}{\text{argmax}} \{ \min_{X_j \in \mathbf{F}} I(X^*, Y | X_j) \}$$

where **F** is the set of already selected features. This tries to ensure that a new feature that is picked is both weakly influenced by the features that have already been picked and is itself important. Running the above steps with the data available, in spite of the simplifications, used up so many resources that I needed to apply some modifications. The features were converted to binary by splitting on their average value, and somewhat analgously to the boosting technique of feature-bagging, the algorithm was applied to mutually exclusive feature subsets whose outputs were then combined. This performed poorly, as discussed below in the Results section.

4.2.2 Perceptron Weights

Because of the way the CMIM algorithm discussed above works, it requires all of the data to be available in memory. This was beyond the computational resources available, hence feature-bagging needed to be used. An alternative and simple feature selection technique that does not require it, is presented here. It is based off the Perceptron algorithm. The advantage of this algorithm is that it works online, meaning that it processes one example at the time, which it then uses to update the weight vector $\mathbf{w}$. The way it does this is through stochastic gradient descent on the following error function:

$$E(\mathbf{w}) = -\sum_{i \in M} y_i(\mathbf{w} \cdot \mathbf{x}_i),$$
$$M = \{\mathbf{x}_i : y_i(\mathbf{w} \cdot \mathbf{x}_i) < 0\}$$

which is derived from the 0/1 loss on the prediction rule:

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x}_i)$$

A Perceptron tries to find a boundary that separates the positive and negative data sets. Unlike an SVM it doesn't find the maximum *margin* hyperplane, but any boundary it will converge to. For linearly separable data it is guaranteed to converge to a separating hyperplane whose parameters will depend on the initialization of $\mathbf{w}$ and the gradient descent step size η. Computing the gradient, and applying gradient descent, gives the following update rule for $\mathbf{w}$:

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \eta \cdot y \cdot \mathbf{x}$$

What the prediction rule shows is that entries of the vector $\mathbf{w}$ that are highest in magnitude are giving importance to the corresponding features in $\mathbf{x}$, and are effectively selecting the relevant information inside of each example. By slightly modifying the algorithm implemented for the CS475 library so that it could truly work 'online', a vector $\mathbf{w}$ was generated for the full training data, and by ranking its entries on magnitude, features corresponding to the largest entries were used to reduce the dimensionality of the data.

This procedure is similar to another algorithm known as SVM Recursive Feature Selection (Guyon at al. 2003), which uses the $\mathbf{w}$ vector of an SVM instead. It was decided not to implement this specific option because the SVM algorithm was already being used to assess the predictive power of selected features. Using it to select features themselves could have introduced unwanted bias towards this algorithm. This way the SVM was kept an unbiased decider of good performance.

As shown in the Results section below, using Perceptron weights performs surprisingly well for being such a simple procedure.

5 Results

5.1 CMIM

As shown in Table 1 below, CMIM doesn't perform better than random for any of the three dimensionality reductions that were attempted. Therefore the features selected by this algorithm were not considered for biological analysis.

5.2 Pereceptron Weights

Because of the high accuracy against random that perceptron weights feature selection shows for a dimensionality reduction as low as 100 features, it is reasonable to believe that some of the selected features have captured biologically significant traits. In Table 2 the top 10 predictive fea-

Feature Number	**Perceptron w**	**CMIM**	**Random**
10,000	72.73%	75.32%	79.63%
1,000	74.48%	76.89%	74.90%
100	75.15%	43.13%	53.78%

Table 1: Results

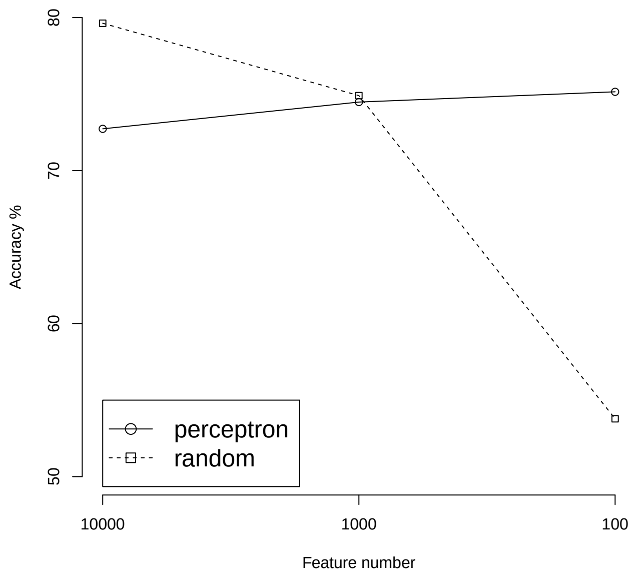

Figure 1: Accuracy of perceptron weights (solid) and random feature selection (dashed)

tures, meaning the ones associated with the 10 highest entries in **w**, are shown together with some matching trascription factors that seem to confirm their significance. Note that the features are pairs of 5-mers, and as described in the Feature Extraction section, correspond to the count of these pairs appearing together in a window of 100 bp.

Feature	Database family match	Top matched transcription factors
TCTGT - TCTCT	HLH	TAL1:TCF3, TCF3, ZFP238
GAGAG - TTTTT	SOX	SOX4, SOX11, SOX10
TGTCT - TCTCT	HLH	TAL1:TCF3, TCF3, ZFP238
GGAAC - GTACT	–	–
GAATT - CTTCT	Homeodomein	LHX3, OTP, PRRX2, PROP1
CAGCT - TTTGT	HLH	NHLH1, ASCL2, HEN1, REPIN1
CGACT - CATTA	Homeodomein	LHX3, OTP, PRRX2, PROP1
GGTAC - GAAAT	–	–
GCAGA - TTGAT	SOX	SOX5, SOX9, SOX10
TCTTT - TAATT	Homeodomain	HOXA6, HOXC8, PRRX2

Table 2: TFs matching top selected features

6 Discussion

The poor performance of CMIM can be attributed most likely to the sparsity of the feature vectors. Unfortunately reducing the features to binary by using their average value for splitting causes a lot of information loss and the algorithm fails to be able to distinguish between the mutual information of different features. It also relies on sequentially selecting features among the entire training data. Using feature-bagging may have reduced its effectiveness as well.

On the other hand, using perceptron weights proves to be effective. It is interesting to observe the trend of test accuracy as the number of selected features is decreased (Figrue 1). While the behavior of random feature selection is as expected, the perceptron accuracy increases for a smaller number of features. The most likely reason for this behavior is that the SVM is overfitting the training data for a large number of features. As they are reduced, while the random set does not contain most of the truly predictive features and therefore its accuracy drops, the perceptron selected set keeps predictive features thereby keeping a high test accuracy.

It's important to notice that the test accuracy achieved for this reduced feature set (75.15%) is lower then the accuracy achieved with the full set (80.55%). It is therefore unclear if the selected subset does now contain spatially interesting features, or if it implicitly contains enough k-mer counts so that the test accuracy is only a consequence of their description of DNA. This is a consequence of the fact that having a count for some feature f corresponding to two k-mers, means that we also implicitly have a count for each single k-mer. The hope was that including the fact that they were appearing together within a spatial constraint could have added useful information, possibly implying some biological structure or function.

To check if this was the case, the data set resulting from only 5-mers counts was reduced using all the 5-mers appearing in the perceptron reduced feature set. This resulted in a reduction from 512 ($4^5 \div 2$) [1] to 200 features (each one of the percetron selected feature was a 'spatial' feature with two 5-mers). SVM test accuracy for this set was 69.45%, which suggests that the spatial component was indeed relevant, and may contain important biological information.

The table above shows how some of the highest scoring spatial 5-mers match known transcription factors. On one hand this gives confidence that both the SVM and the perceptron are picking up on some tangible biological structures. On the other it now allows to infer something about the mode of interaction between transcription factors at a certain distance from each other and larger proteins, like P300, involved in enhancer binding.

6.1 Future Work

The purpose of this project was to see if it was possible to find a simple spatial description of enhancer sequences that could be shown to add information to the already used k-mer framework. Now that this seems possible, the next step is to refine both the feature extraction and the feature selection techniques presented in this project. Examples would be to include precise base pair distances between k-mers and not just a window size in which they appear together. Trying other feature selection techniques like SVM-RFE or finding a better implementation for CMIM. The goal that could be achieved eventually is to be able to state which transcription factors bind P300 and other enhancer protein, in what sequence and in which biological tissues. How polymorphism linked to Alzheimer's and other diseases locate in these TF binding sites and what binding interactions do they alter. This could explain why some people that present with these polymorphisms never develop a disease while others do. Such differences may be at the bases of developing effective treatments, which so far has been difficult, due to a lack in understanding of the dynamics of these diseases.

These are exciting times in genomic research.

[1]Because of the double stranded structure of DNA, two sequences can be considered equivalent when they are 'mirrored' (GCAT = ATGC). Hence the division by two.

7 Comparison to Proposal

I managed to do most of what I set out to do in the proposal. I didn't implement SVM-RFE both because I ran out of time, and because I didn't like using SVM for both classification and feature selection (as explained above). One thing that I had hoped to get to but I didn't have enough time for, was comparing some of the selected features with some known enhancer complexes, to see if their spatial distribution matched the positioning of TFs.

References

Banerji J. (1981) Expression of a β-globin gene is enhanced by remote SV40 DNA sequences *Cell*,**27**:299-308

Bishop, C.M. (2011). *Pattern Recognition and Machine Learning*,Springer Science.

Guyon, I.; Elisseeff, A.(2003) An Introduction to Variable and Feature Selection.
Journal of Machine Learning Res. **3**

Joachims,T. (1999) Making Large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*, B. Schlkopf, C. Burges, A. Smola (ed.). MIT-Press.

Lee D, Karchin R, Beer M.A. (2011) Discriminative prediction of mammalian enhancers from DNA sequence. *Genome Res.* **21**:2167-2180

Matys V, Fricke E, Geffers R, Gossling E, Haubrock M, Hehl R, Hornischer K, Karas D, Kel AE, Kel-Margoulis OV, *et al.* (2003) TRANSFAC(R): Transcriptional regulation, from patterns to profiles. *Nucleic Acids Res.*, **31**:374-378

The 1000 Genomes Project Consortium. *et al.* (2010) A map of human genome variation from population-scale sequencing. *Nature*, **467**:1061-1073

Visel A, Blow MJ, Zhang T, Akiyama JA, Holt A, Plajzer-Frick I, Shoukry M, Wright C, Chen F, Afzal V, *et al.* (2009) ChIP-seq accurately predicts tissue-specific activity of enhancers *Nature*, **457**:854-858

Wang G., Lochovsky F.H., Yang Q. (2004) Feature selection with conditional mutual information maximin in text categorization *Proceedings of the 13th ACM International Conference on Information and Knowledge Management, ACM, Washington, DC, USA*:342-349