Relevance Vector Machines and Eponine Models for Genome Sequence Analysis

Konstantin Tretyakov (kt@ut.ee)

December 20, 2005

Abstract

Relevance vector machines (RVM) is a family of machine learning methods, introduced by Tipping, that represent a bayesian approach to the training of general linear models (GLM). RVM-s are reported to be able to effectively produce convenient sparse representations of data thus competing with the popular support vector machines. When used with a suitable set of basis functions, RVM-s can be applied to the analysis of genome sequence data. The Eponine models provide such sets of functions. This article gives a brief (and probably slightly simplistic) overview of the application of the RVM with Eponine to genome sequence analysis, a work originally described by Down.

1 Introduction

Machine learning methods play an important role in bioinformatics by providing ways of inferring useful knowledge from raw data. Methods like hidden markov models, neural networks, support vector machines, etc. have their steady place in the arsenal of common pattern analysis tools of bioinformatics. These methods are used in a multitude of contexts such as *classification* of genes and DNA sequences or *detection of motifs* identifying *transcription start sites (TSS)* or *transcription factor binding sites (TFBS)*. This article presents a brief overview of yet another family of machine learning methods, the *Relevance Vector Machine (RVM)*, which, when combined with the *Eponine* model, can be used to classify DNA sequences and to infer useful motif information.

The RVM method was first introduced by Tipping [1]. The Eponine set of models and their application to DNA sequences is due to Down [3, 4, 5, 6]. This work aims to give a very brief summary of the cited papers. The basic context is explained first. Then follows the description of relevance vector machines and the Eponine family of models, illustrated by the applications to sequence analysis.

2 Preliminaries

One of the central questions in bioinformatics is related to the analysis of DNA sequences of the genes, of their promoter regions and even of the non-coding regions of the genome. One particularly common and promising method is learning *classification models* for the sequences into several classes. As an example, consider two sets of DNA sequences—P^+ and P^-—such that all the sequences s in P^+ are taken from the promoter regions of certain genes, and the sequences in P^- are just random parts of the genome. Suppose that we shall be able to somehow find a *classifier* function f, that has $f(s) = 1$ for $s \in P^+$ and $f(s) = 0$ for $s \in P^-$. There are two important applications for such a function. Firstly, we may assume that the function *generalizes*, i.e. it would mostly return 1 when given a substring of *any* gene promoter, and 0 otherwise. In this case we can use the function to classify previously unseen sequences and thus find other promoter regions in the genome. Secondly, we may try to "look inside" f, and attempt to understand *how* the classifier makes its decision and thus *what* are the features discriminating P^+ from P^-. Clearly, in order to use this approach we need a method for constructing classifiers from data, that is, we need *machine learning.*

Machine learning is a discipline, the general task of which is often viewed as inferring an unknown relationship $f : X \to Y$ from a *training set* of known *samples* $(x_n \to y_n), n \in \{1, \dots, N\}$. It is most often done by fixing the general parameterized form of the relationship f (i.e. specifying the *model*) and using the available data to find the *parameters* (also called *weights*) of the model to make it *fit* the data. One particularly useful class of models is the class of *generalized linear models (GLM)*. A GLM is a model of the form:

$$f_{\mathbf{w}}(x) = \sum_{m=1}^{M} w_m \phi_m(x) + w_0$$

where ϕ_m is a set of M *basis functions* (which can be arbitrary real-valued functions) and $\mathbf{w} = (w_0, w_1, \dots, w_M)^T \in \mathbb{R}^M$ is a vector of weights that will be adjusted. A notable feature of this model is that the input variable x enters the formula only via the basis functions ϕ_m, and thus in theory the model can be applied to any type of data X, for which suitable basis functions ϕ_m can be provided. The resulting model is a function $f : X \to \mathbb{R}$ and may thus be applied to certain regression tasks. As in our case we are considering classification, we shall use the following variant of the model:

$$f_{\mathbf{w}}(x) = \sigma\left(\sum_{m=1}^{M} w_m \phi_m(x) + w_0\right) \tag{1}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the *logistic function*, that rescales the output such that $0 \le f_{\mathbf{w}}(x) \le 1$ and can be interpreted as a probability that given x is a member of the "positive" class.

There are two questions that remain to be answered before we are able to apply the model to genome-sequence data:

- How do we infer the suitable values for the parameters $\mathbf{w}$ from the data?
- What are the functions ϕ_m?

We shall use the Relevance Vector Machine method for finding the parameters and the Eponine model for the basis functions.

3 Relevance Vector Machine

The Relevance Vector Machine (RVM) is a *sparse* method for training generalized linear models. This means that in general, many of the inferred parameters w_m will be zeroes. Sparsity is often a desirable feature because it implies simplicity, effectiveness and better interpretability of the resulting model. The RVM is named by analogy to the better-known Support Vector Machine method, which also results in sparse models.

The RVM is a bayesian approach to finding the parameters, that is, it attempts to find the vector of parameters that is most probable, given the available data. Let $\mathbf{x} = (x_1, \dots, x_N)$ denote the list of all input sequences and $\mathbf{t} = (t_1, \dots, t_N)$ the corresponding class labels ($t_n \in \{0, 1\}$). To find the parameter vector $\mathbf{w}$ we shall maximize $P(\mathbf{w} \,|\, \mathbf{x}, \mathbf{t})$ which can be rewritten using the Bayes' formula as:

$$P(\mathbf{w} \,|\, \mathbf{x}, \mathbf{t}) = \frac{P(\mathbf{x}, \mathbf{t} \,|\, \mathbf{w}) P(\mathbf{w})}{P(\mathbf{x}, \mathbf{t})} \propto P(\mathbf{x}, \mathbf{t} \,|\, \mathbf{w}) P(\mathbf{w}) \tag{2}$$

where the constant factor in the denominator does not matter because we are maximizing wrt $\mathbf{w}$. An appropriate way to compute $P(\mathbf{x}, \mathbf{t} \,|\, \mathbf{w})$ is

$$P(\mathbf{x}, \mathbf{t} \,|\, \mathbf{w}) = \prod_{n=1}^{N} f_{\mathbf{w}}(x_n)^{t_n} (1 - f_{\mathbf{w}}(x_n))^{1-t_n} \tag{3}$$

where $f_{\mathbf{w}}$ is given by (1). The distribution $P(\mathbf{w})$ can be specified in any way, as it should merely reflect our *prior belief* in what are the likely values for the weights. As we are interested in getting a model having lots of weights close to zero, it is reasonable to choose some decently narrow zero-mean distribution as a prior. The RVM method uses the *Automatic Relevance Determination* principle and describes the distribution of each weight w_m by a normal distribution with mean 0 and variance α_m^{-1} . That is,

$$P(\mathbf{w} \,|\, \boldsymbol{\alpha}) = \prod_{m=1}^{M} \mathcal{N}(w_m \,|\, 0, \alpha_m^{-1}) \tag{4}$$

where $\mathcal{N}(w_m \,|\, 0, \alpha_m^{-1}) = \frac{1}{\alpha_m^{-1}\sqrt{2\pi}} e^{-w_m^2/2\alpha_m^{-2}}$ is the gaussian density function. Now that we have introduced new parameters α_m we must define priors $P(\boldsymbol{\alpha})$ for them too (so-called *hyperpriors*) because

$$P(\mathbf{w}) = \int P(\mathbf{w} \,|\, \boldsymbol{\alpha}) P(\boldsymbol{\alpha}) d\boldsymbol{\alpha} \tag{5}$$

For RVM, a very broad gamma distribution is often used, but the exact choice does not matter much as long as the distribution is wide enough. The very trick of introducing individual hyperpriors for all the weights is the key feature of the model and is ultimately responsible for its sparsity properties. By combining equations (3), (4) and (5) we obtain the expression for $P(\mathbf{w}\,|\,\mathbf{x},\mathbf{t})$ that needs to be *maximized* in order to find the suitable weights.

The expression is too complicated to be optimized analytically, so an iterative procedure based on MacKay [7] is usually utilised:

1. For current, fixed values of $\boldsymbol{\alpha}$, find the most probable weight vector $\mathbf{w}_{MP}$. This is equivalent to the standard optimization of a regularized logistic regression model and can be done using the efficient *iteratively reweighted least-squares* algorithm [8].

2. Assume that $P(\mathbf{w},\mathbf{x},\mathbf{t}\,|\,\boldsymbol{\alpha})$ is approximately a multidimensional normal distribution centered at $\mathbf{w}_{MP}$, estimate its covariance matrix at $\mathbf{w}_{MP}$ and use this approximation to update the $\boldsymbol{\alpha}$ parameters. The details of this step require a bit too much discussion to be included in this report, so the interested reader is referred to [2] for an in-depth description.

The procedure is repeated iteratively until suitable convergence criteria is met. Note that the algorithm is similar in spirit to the expectation-maximization algorithm. In fact, EM could be applied directly to the maximization of $P(\mathbf{w}\,|\,\mathbf{x},\mathbf{t},\boldsymbol{\alpha})$ with $\boldsymbol{\alpha}$ being the hidden variables.

The algorithm posesses an important property: during the iterations many of the parameters α_m will often grow to infinity, which means that the corresponding weights w_m are with high probability equal to zero, and can be completely discarded from the model. This way, the working set of *relevant* basis functions ϕ_m quickly decreases until a sparse final solution is found.

Now, in order to apply the RVM algorithm to sequence data we only need to find the appropriate basis functions ϕ_m.

4 Eponine RVM Models

4.1 Position Weight Matrices

The basic element of all the Eponine models is the *position weight matrix (PWM).* A PWM W_{ij} is a two-dimensional array of positive real numbers of size $4 \times K$, that is indexed by alphabet characters in one dimension ($i \in \{\mathrm{A},\mathrm{C},\mathrm{T},\mathrm{G}\}$) and integer *positions* in the second dimension ($j \in \{1,\ldots,K\}$). A PWM can be *matched* to a string s of length K. The result of a match, denoted by $W(s)$, is a real number computed as

$$W(s) = \prod_{k=1}^{K} W_{s_k k}$$

It is often the case that each value W_{ij} of a PWM represents the probability of having the character i at position j in a sequence. In this case a PWM can be viewed as a probabilistic model for sequences of length K. It is often convenient to take logarighms of the values in the PWM. The matching procedure then takes the form of a sum rather than a product.

4.2 Eponine Anchored Sequence

The *Eponine Anchored Sequence (EAS)* is a system for analyzing the genomic sequence around a certain *anchor point.* A number of important sequence analysis questions, such as prediction of transcription start sites or splice sites, can be phrased in terms of analyzing sequence points. The system takes an *anchored sequence* as its input. That is a DNA sequence s coupled with an index a specifying the position that should be analyzed. For example we might use the anchored sequence $(\texttt{ATGCATGGCG}, 3)$ when asking, whether the third character in it (`G`) is the transcription starting point for some gene. The model's output would be a large positive real number if it is probably true, and a large negative number if it is probably false. The model is parameterized by a *positioned constraint (PC).* A positioned constraint is a pair consisting of a *position-weight-matrix* W and a *probability distribution* P *over integer offsets.* The model *scores* a given anchored sequence (s,a) as:

$$\phi_{W,P}(s,a) = \frac{1}{|W|} \log \sum_{i=-\infty}^{\infty} P(i) W(s[a+i, |W|])$$

where $|W|$ is the number of columns in W, and $s[a+i,|W|]$ is the substring of s of length $|W|$ starting from index $a+i$. As a simple example, suppose that W is a PWM describing a single sequence `GATGAG`, and P is a distribution that is nonzero only at $i = -3$. The corresponding EAS $\phi_{W,P}$ will then assign high scores only to the anchored sequences that have the string `GATGAG` starting three positions before the anchor point.

EAS provides us with an infinite set of functions $\phi_{W,P}$. We may build a linear combination of a subset of such functions and use the RVM algorithm to find the weights that enable the model to, say, detect transcription start sites. Once this is done, we might examine the positioned constraints of the relevant functions that were left after training. It is probable that these describe important motifs in the sequences.

The drawback of the approach is the fact that we have to select a finite subset of functions $\phi_{P,W}$ to run the algorithm upon. Most probably the subset will either be too small to contain relevant functions, or too large for the training to be feasible. The problem can be alleviated by generating new functions on-the-fly, an approach called *sampling RVM.*

4.3 Sampling RVM

Sampling RVM is an approach that allows the RVM algorithm to converge on a small finite subset of relevant basis functions from a very large (potentially infinite) set of all possible functions. The idea is the following: we start running the RVM algorithm on an *initial subset* of basis functions of size I. During the course of the algorithm the irrelevant functions get pruned away and the size of the working set decreases. Once it hits the *low water mark* of L functions, we increase the size of the working set to *high water mark* H by adding new functions from the pool of yet unused ones, and restart training. This way new functions enter the algorithm incrementally, thus trading some computation time for memory.

When the whole set of basis functions is infinite it makes sense to use heuristics when generating new functions to be added to the model. For example we can generate new functions so that they are in some way "close" to those already in the working set. In the case of EAS, that would mean that the PWMs and offset distributions of the newly generated functions are slightly perturbed versions of the ones from the working set. The idea thus resembles a genetic algorithm and is based on the observation that if a function ϕ is relevant then another function close to it will also probably be relevant.

4.4 EAS Applications

The EAS model together with the Sampling-RVM algorithm has been applied to the task of transcription start site prediction. The input data was taken from the EPD database. The positive class consisted of the real transcription start site anchor points, and the negative class contained random points in the genome. The trained model consisted of 4 basis functions, one of which indicated the well-known `TATA`-box at offset -30 from the site, and the three others favoured `CG`-rich regions. This is consistent with the current knowledge, and the model's predictive performance was comparable to existing methods such as *PromoterInspector* or *CpG.*

4.5 Eponine Windowed Sequence

The *Eponine Windowed Sequence (EWS)* model is similar to EAS, but is used to analyze whole sequences rather than points. The EWS consists of a single PWM W, that is scanned over the whole input sequence to give the score:

$$\phi_W(s) = \frac{4^{|W|}}{|s| - |W| + 1} \sum_i W(s[i, |W|])$$

Analogously to the case of EAS we can construct a linear combination of functions ϕ_W and use the sampling RVM method (with the genetic-algorithm-kind of generation strategy) to infer a simple classifier model containing a small number of them. Analogously, the PWM-s of the functions in the inferred model should have some biological significance. The EWS model has been successfully trained on windows of sequence upstream of annotated genes in which case it worked successfully as a promoter predictor.

A useful extension of EWS is the *convolved EWS (C-EWS)*, that can capture large-scale patterns in sequences. In this case, the basis function is not a single PWM, but a "scaffold" consisting of one or more PWM-s, each with associated position distribution relative to the scaffold's anchor. Suppose $\mathcal{W} = (W_1, \dots, W_K)$ are PWM-s, and $\mathcal{P} = (P_1, \dots, P_K)$ their corresponding position distributions. The score of a C-EWS $\phi_{\mathcal{W},\mathcal{P}}$ is calculated by scanning the scaffold over the whole sequence:

$$\phi_{\mathcal{W},\mathcal{P}}(s) = Z \sum_{i=-\infty}^{\infty} \left(\prod_{k=1}^{K} \left(\sum_{j=-\infty}^{\infty} P_k(j) W_k(s[i+j, |W_k|]) \right) \right)$$

where Z is the normalizing constant:

$$Z = \frac{4^{\sum_{k=1}^{K} |W_k|}}{C}$$

where C is the number of addends in the first sum over i (we do not really need to sum from $-\infty$ to ∞, but only over those indices for which some part of the scaffold intersects the sequence with nonzero probability).

Again, C-EWS can be used with the sampling RVM to infer useful models. Due to model's complexity, it might make sense to explore only small scaffolds of up to 3 PWM-s. The C-EWS algorithm has been applied to detect exonic splice sites.

5 Summary

Relevance Vector Machines is an interesting sparse machine learning method, that, when combined with the Eponine sequence models, can be applied to extract useful motif information from genome sequences. RVM training can be done in a reasonably effective way using an iterative algorithm similar in spirit to EM. The results of applying the RVM+Eponine combination to TSS prediction, promoter prediction and exonic splice site detection are promising.

References

[1] M. Tipping. The relevance vector machine, 2000.

[2] Michael E. Tipping. Sparse bayesian learning and the relevance vector machine, 2001.

[3] Thomas A Down and Tim J P Hubbard. Computational detection and location of transcription start sites in mammalian genomic DNA. *Genome Res*, 12(3):458–61, Mar 2002.

[4] Thomas A. Down and Tim J. Hubbard. Relevance vector machines for classifying points and regions in biological sequences, 2003.

[5] Thomas A Down and Tim J P Hubbard. What can we learn from noncoding regions of similarity between genomes? *BMC Bioinformatics*, 5:131, Sep 2004.

[6] Thomas A. Down, Bernard Leong, and Tim J. Hubbard. A machine learning strategy to identity exonic splice enhancers in human protein-coding sequence, March 2004.

[7] D.J.C. Mackay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

[8] I.T. Nabney. Efficient training of RBF networks for classification. In *Proceedings of ICANN99*, pages 210–215, 1999.