Gradient-Based Feature Selection for Conditional Random Fields and Its Applications in Computational Genetics

Minmin Chen Yixin Chen Michael R. Brent Aaron E. Tenney
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis, Missouri, USA
{mc15, chen, brent, aet1}@cse.wustl.edu

Abstract

Gene prediction is one of the first and most important steps in understanding the genome of a species, and different approaches haven been proposed. In 2007, a de novo gene predictor, called CONTRAST, based on Conditional Random Fields (CRFs) is introduced, and proved to substantially outperform previous predictors. However, the oversize feature set used in the model has posed several issues, like overfitting problem and excessive computational demand. To resolve these issues, we did a thorough survey of two existing feature selection methods for CRFs, namely the gain-based and gradient-based methods, and applied the later one to CONTRAST. The results show that with the gradient-based feature selection scheme, we are able to achieve comparable or even better prediction accuracy on testing data, using only a very small fraction of the features from the candidate pool. The feature selection method also helps researchers better understand the underlying structure of the genomic sequences, further provides insights of the function and evolutionary dynamics of genomes.

1 Introduction

Gene prediction is one of the most extendedly studied problems in computational biology, and different approaches have been proposed, such as GENESCAN [3], CRAIG [2], TWINSCAN [11], and N-SCAN [6]. Most of these gene predictors are based on Hidden Markov Models (HMMs). However, the independent assumptions made in HMMs greatly limit their performance. To overcome the limitations of HMMs, Lafferty et al. introduced Conditional Random Fields (CRFs) [12], which allow complex, overlapping, and inter-dependent features to be included to aid learning. In 2007, a de novo gene predictor based on CRFs, CONTRAST (CONditionally TRAined Search for Transcripts), is proposed. By including features effectively modeling the dependecies between multiple informants, CONTRAST is able to show substantial improvement over previous de novo gene predictors.

Although the work has shown great success, there exist potential improvements. We believe, one of the most significant extensions will be feature selection. The original model contains 33,003 features. Though a rich feature set can help mine data, too many features can be detrimental. In general, as more features, especially irrelevant or redundant ones, are included in a model, the model will have greater tendency to overfit the training data. Also, a large quantity of features bring about greater demands on computational costs, including training/testing time and storage space. Moreover, learning a "good" model can be an important task in its own right, as it can provide insights about the underlying structure of the domain. By feature selection, we aim to select features bearing important information, while discarding redundant ones. Therefore, to resolve the issues.

Desirable properties of a feature selection method include the following:

a) The size of the selected subset should be significantly smaller than the full candidate feature set;

b) The selected subset should reflect domain knowledge;

c) The selected subset derived from a data set should generalize well to other statistically similar datasets;

A great number of feature selection methods have been introduced in the last decades, like filtering approaches [9, 19, 4], wrappers approaches [10], and embedded approaches [17, 5, 8]. Although feature selection has been extensively studied for other data mining models, practical and robust algorithms that select features for CRFs are still few. One feature selection method for CRFs, which we call *gain-based* method, falls into the embedded method paradigm [16, 14]. At each iteration, the feature with the maximum potential gain is added to the feature set. The gain of a candidate feature is measured by the maximum reduction of the objective function value that can occur by adjusting the weight of the candidate feature while keeping the weights of other features fixed. In 2003, Perkins et al. [15] proposed a feature selection algorithm based on gradient information, which we call *gradient-based* method, for a linear model. In 2006, Lee et al. [13] combined the two feature selection methods with L_1,L_2 regularization respectively, to learn the underlying structure for CRFs. However, as the work focus on comparing the two regularization methodologies, the effects of the two feature selection methods are ignored instead. To the best of our knowledge, there exists no work comparing these two important feature selection methodologies against each other, neither to other feature selection approaches.

In this paper, we did a thorough survey of these two feature selection methods, also compared them to one filtering approach. As we will discuss later, the gradient-based approach turns out to be much more effective and efficient on avoiding overfitting and reducing the computational costs. Another nice property of the gradient-based approach is that the gradient in CRFs has intuitive meanings that give a satisfactory explanation to the validation of this approach. We applied the gradient-based feature selection algorithm to CONTRAST. The experimental results show that the gradient-based approach is very attractive and gives better performance than gain-based method when evaluated against the criteria mentioned before.

This paper is organized as follows. Section 2 gives a brief review of CRFs and Section 3 presents a detailed comparison of the gain-based and gradient-based algorithms. Section 4 reports experimental results of applying the gradient-based feature selection to a state-of-the-art gene predictor and performing gene prediction on fly genomes. We give conclusions in Section 5.

2 Conditional Random Fields

A CRF considers the distributions over a set of random variables $V = X \cup Y$, where X is the set of variables over the observation sequences to be labeled (e.g., a DNA base sequence), and Y is the set of random variables over the corresponding labeling sequences (e.g., the labeling of gene sites). It models the conditional distribution of the labeling sequence given the observation sequence based on the set of *feature* functions. Each predefined *feature* $f_k(\mathbf{y}_t, \mathbf{x}_t)$ returns a numerical value for any configuration of $\mathbf{y}_t$ and $\mathbf{x}_t$.

In this paper, we focus on linear-chain CRFs, a subclass of CRFs that is computationally tractable and widely used. In a linear-chain CRF, each *feature* involves only two consecutive hidden states. Let **y** and **x** be the labeling and observation sequences, respectively, $\{f_k(y_t, y_{t-1}, \mathbf{x}_t)\}_{k=1}^K$ a set of real-valued feature functions, and $\Lambda = \{\lambda_k\} \in \mathcal{R}^K$ the weight vector, the distribution over the label sequence **y** given **x** in a linear-chain CRF takes the form

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}, \quad (1)$$

where Z(**x**) is an instance-specific normalization function

$$Z(\mathbf{x}) = \sum_{\mathbf{y'}} \exp\left\{\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y'_t, y'_{t-1}, \mathbf{x}_t)\right\}.$$

Given training data $\mathcal{D} = \left\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\right\}_{i=1}^N$, to estimate the weights $\Lambda = \{\lambda_k\}$, we typically minimize the negative conditional log likelihood:

$$\text{minimize}_\Lambda \quad \mathcal{O}(\Lambda) = -\sum_{i=1}^{N} \log p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}). \quad (2)$$

3 Gain-based vs. Gradient-based Feature Selection

In this section, we present a comprehensive study of the gain-based feature selection scheme and the gradient-based one. An intuitive explanation of why the gradient-based feature selection scheme works better for CRF is given as well.

3.1 Framework

Both schemes fall into the embedded method paradigm, which iteratively construct a feature set by adding one feature to the set at each iteration according to certain criterion. As shown in Figure 1, each iteration has two steps, including a feature selection step (shown in dashed) and a parameter estimation step (shown in solid). In the feature selection step, $\beta(\Lambda^\star, \lambda_i)$ measures the potential benefit of including feature i. Let $\mathcal{F}$ be the candidate pool, which contains all the features not yet selected. Let $\mathcal{S}$ be the selected feature subset, which contains the promising features already included in the model so far. For both methods, the weights for features in $\mathcal{S}$ are free variables, while the weights for other features are fixed at 0. At the beginning, we have all the candidate features in the candidate pool $\mathcal{F}$ and the selected subset $\mathcal{S}$ empty. At each iteration, the most discriminative feature according to the criterion β is removed from $\mathcal{F}$ and added to $\mathcal{S}$. Then, a CRF with all the features in $\mathcal{S}$ is re-trained. The process repeats until a regularized CRF objective function can no longer be reduced.

Now the problem is to find a criterion that can identify the features that are strongly discriminative. Since we want a good fit of the CRF model to the training data, it is natural to select features that, when included in the model, will bring the maximum reduction to the objective function. However, computing the exact reduction after adding one

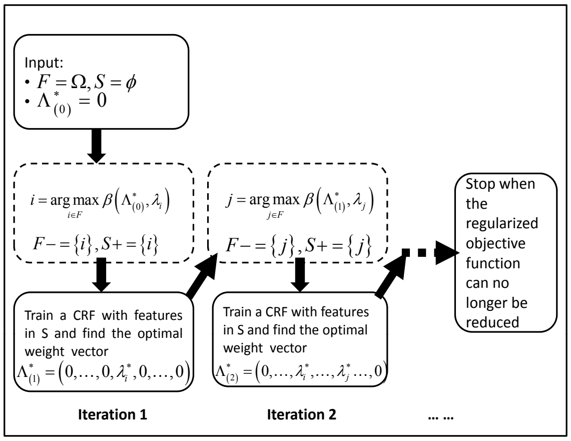

Input:
• $F = \Omega, S = \phi$
• $\Lambda^*_{(0)} = 0$
$i = \arg\max_{i \in F} \beta\left(\Lambda^*_{(0)}, \lambda_i\right)$
$F- = \{i\}, S+ = \{i\}$
$j = \arg\max_{j \in F} \beta\left(\Lambda^*_{(1)}, \lambda_j\right)$
$F- = \{j\}, S+ = \{j\}$
Stop when the regularized objective function can no longer be reduced
Train a CRF with features in S and find the optimal weight vector $\Lambda^*_{(1)} = \left(0, \ldots, 0, \lambda_i^*, 0, \ldots, 0\right)$
Train a CRF with features in S and find the optimal weight vector $\Lambda^*_{(2)} = \left(0, \ldots, \lambda_i^*, \ldots, \lambda_j^* \ldots, 0\right)$
Iteration 1
Iteration 2
… …

Figure 1. The framework of the two feature selection methods.

feature requires re-training the entire CRF model, which is computationally expensive. To overcome this drawback, exiting works aim at approximating the reduction with a computationally scalable measurement.

3.2 Gain-based Feature Selection

The gain-based feature selection scheme measures the significance of each candidate feature by the log-likelihood gain of adding this feature to the model. For each candidate feature f_i, λ_i, $\beta(\Lambda^\star, \lambda_j) = (O)(\Lambda^\star, \lambda_j^\star) - (O)(\Lambda^\star)$. That is, it optimizes the weights of the features already selected first, then finds out the maximum reduction of the objective function value that can occur by adjusting the weight λ_j of the candidate feature while keeping the weights of other features fixed.

A disadvantage of this approach is its exceedingly high computational cost. To assess the gain of each candidate feature will require an optimization process. At each iteration, in order to select the feature with the maximum gain, K optimizations are required, where K is the size of the candidate pool. For many applications, K can be very large and it is very expensive to select even only one feature. Since most problems would require hundreds or even thousands of feature selection iterations in order to find a good feature subset, this approach is not very practical.

3.3 Gradient-based Feature Selection

The gradient-based feature selection method measures the significance of a candidate feature as follows:

Similarly, at each iteration, before a feature selection step is carried out, all the features in the currently selected subset $\mathcal{S}$ have been optimized. Thus, an optimized CRF gives a weight vector Λ^*, where $\lambda_i^* = 0$ for all $i, f_i \notin \mathcal{S}$. We can now approximate the value of the objective function in the neighborhood of Λ^* by

$$\mathcal{O}(\Lambda) \approx \mathcal{O}(\Lambda^\star) + \nabla\mathcal{O}(\Lambda^\star)^T(\Lambda - \Lambda^\star), \tag{3}$$

or, equivalently,

$$\mathcal{O}(\Lambda) - \mathcal{O}(\Lambda^\star) \quad \approx \quad \nabla\mathcal{O}(\Lambda^\star)^T(\Lambda - \Lambda^\star). \tag{4}$$

Now suppose we add feature f_j to $\mathcal{S}$ and perturb the value of λ_j while keeping other values in Λ^* unchanged, the change of objective function value can be approximated by

$$\begin{aligned} & \mathcal{O}(\Lambda) - \mathcal{O}(\Lambda^\star) \\ \approx & \sum_{i|f_i \in \mathcal{S}} \frac{\partial\mathcal{O}(\Lambda^\star)}{\partial\lambda_i}(\lambda_i - \lambda_i^\star) + \frac{\partial\mathcal{O}(\Lambda^\star)}{\partial\lambda_j}(\lambda_j - \lambda_j^\star) \\ = & \frac{\partial\mathcal{O}(\Lambda^\star)}{\partial\lambda_j}(\lambda_j - \lambda_j^\star), \end{aligned} \tag{5}$$

where the last equality is true since $\frac{\partial\mathcal{O}(\Lambda^\star)}{\partial\lambda_i} = 0$ for any $i, f_i \in \mathcal{S}$ when Λ^* is the optimal weight vector for features in $\mathcal{S}$.

From (5) we see that, if we let the step size $\lambda_j - \lambda_j^\star$ be the same for each candidate feature $j \in \mathcal{F}$, the feature with the maximum absolute derivative value $|\frac{\partial\mathcal{O}(\Lambda^\star)}{\partial\lambda_j}|$ will bring the maximum reduction to the first-order approximation of the objective function, since we have the freedom to set λ_j to be positive or negative. That is, by adding that feature into the model, the objective function $\mathcal{O}(\Lambda)$ will decrease the fastest. In summary, the absolute value of the derivative is a measurement of the sensitivity of a model with respect to each candidate feature. A large derivative is an indication of a possible large reduction of the objective function locally if the candidate feature is selected.

Recall that, in CRF, the derivative of $\mathcal{O}$ with respect to feature f_j is:

$$\begin{aligned} \frac{\partial\mathcal{O}}{\partial\lambda_j} = & \sum_{i=1}^{N}\sum_{t=1}^{T} f_j(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) \\ & - \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{y,y'} f_j(y, y', \mathbf{x}_t)p(y, y'|\mathbf{x}^{(i)}). \end{aligned} \tag{6}$$

A satisfactory interpretation of the derivative in (6) is as follows. The first term is the **empirical count** of feature j in the training data, and the second term is the **expected count** of this feature under the current trained model. Hence, the derivative measures the difference between the empirical count and the expected count of a feature under the current model.

A small derivative implies that the current model has largely captured the information contained in this feature and adding this feature to the model will not help much. On the other hand, a large derivative indicates a large difference between these two terms. Suppose that in the training data a feature f_j appears R times, while under the current model, the expected count of f_j is M. We would expect that $|R - M|$ positions where f_j appears are mistakenly labeled under the current model. When $|R - M|$ is large, adding f_j to the model will help the model correct many mistakenly labeled positions, justifying the reasonable-ness of the gradient based method.

For CRF, the derivatives of all the features can be directly calculated with one run of the forward-backward algorithm [18], which is much more efficient than optimizing a CRF model K times, as required by gain-based feature selection scheme.

3.4 Comparison

In this section, we presents a detailed comparison of the effectiveness of both methods on addressing overfitting and learning a good model.

The experiments were carried out on a toy casino gambling example shown in Figure 2. In the model, there is a real die with an equal probability of rolling each number, and a fake die with a much higher probability of rolling the number 6. The generated training and testing sequences both contain 30,000 data points.

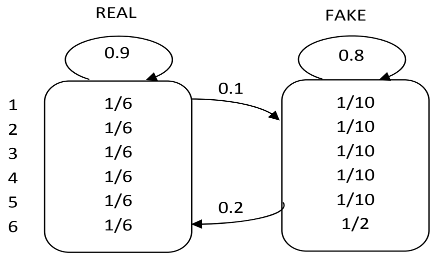


Figure 2. A toy casino gambling model.

Given the training data, we constructed five different feature sets $\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4$, where each set $\mathcal{F}_i$ contains four transition features $T(real, real)$, $T(real, fake)$, $T(fake, real)$, $T(fake, fake)$, and up to the i^{th} order emission features. For example, $\mathcal{F}_0$ has only 0^{th} order emission features $E(y_t, x_t)$, such as $E(real, 1)$, while $\mathcal{F}_1$ has 1^{st} order emission features $E(y_t, x_t, x_{t+1})$, such as $E(real, 11)$. Thus, we have $\mathcal{F}_0 \subset \mathcal{F}_1 \subset \mathcal{F}_2 \subset \mathcal{F}_3 \subset \mathcal{F}_4$, and the size of the feature sets grows exponentially. We trained CRFs with different feature sets. The experimental results show that a large feature set not only causes overfitting, but also prohibitively large computational costs. Due to space limitation, the results are not presented here.

Since gain-based feature selection is not very practical for tasks with large candidate pool, we carry out the comparison on two relatively small candidate feature sets: $\mathcal{F}_0$ with $K = 16$, and $\mathcal{F}_1$ with $K = 88$. To avoid any possible side effects, we do not include a regularization term in this experiment. Instead, we let both methods run until all the candidate features are selected. For both methods, whenever a new feature is added to the model and retrained, we plot the prediction accuracy of the trained model on unseen testing data in Figure 3.

The solid lines in Figure 3(a) and 3(b) stand for the prediction accuracy of the trained CRF model built with all the features in $\mathcal{F}_0$ and $\mathcal{F}_1$, respectively. As shown in Figure 3(a) and 3(b), for both candidate sets, the gradient-based method can identify a smaller feature subset that gives the same accuracy as the full feature set, compared to the gain-based method.

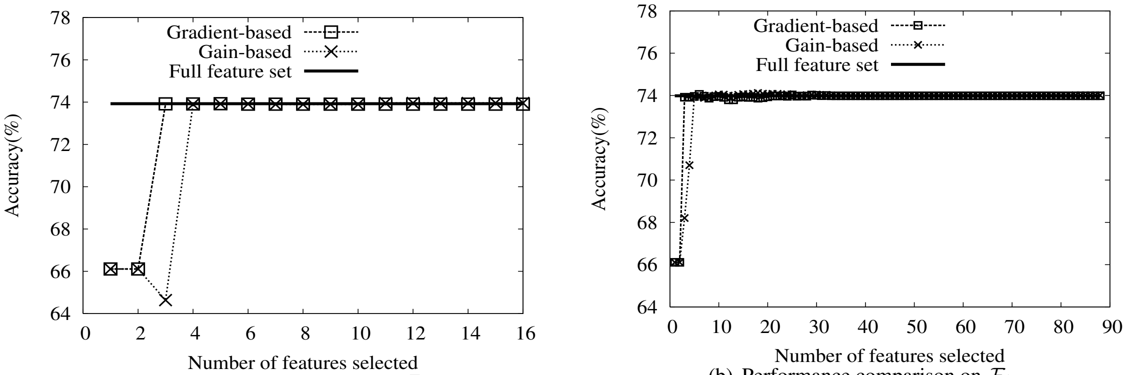
As shown in Figure 3(a), for candidate feature set $\mathcal{F}_0$, instead of using all the 16 features in $\mathcal{F}_0$, the gradient-based feature selection is able to reach the same prediction accuracy after selecting only three features. On the other hand, the gain-based method requires four features to reach the same accuracy. Moreover, the third feature selected by the gain-based method actually deteriorates the performance of the model greatly.

A similar phenomenon can be observed for the candidate feature set $\mathcal{F}_1$, as shown in Figure 3(b). The model with the subset selected by gain-based method, which has only six features, actually outperforms the one with all the 88 features included. In contrast, the gain-based method converges at eight features.

Table 1 and Table 2 show detailed comparisons of the time, reduction of objective function, prediction accuracy, and selected feature of each method at each iteration.

The most significant contribution of the gradient-based method is that it avoids the massive computational time demanded by the gain-based method. It is several orders of magnitude faster than the gain-based method. As we can see from the tables, the time the gain-based method takes to select a feature grows exponentially as the candidate pool grows. As shown in Table 2, for candidate pool $\mathcal{F}_1$, at each iteration, while it takes around 20 minutes for the gain-based method, the gradient-based method is able to pick out the most promising feature within one second. When applying the gain-based method on a candidate pool with only 88 candidate features, the whole feature selection procedure takes around 36 hours to finish, while the gradient-based method takes only around 5 minutes. The computational time savings of the gradient-based method will be even more significant when the candidate pool is larger.

In addition to being much more efficient, the gradient-based method also achieves better accuracy on testing data than the gain-based method with less features. Most of the times the gradient-based method selects a feature that leads to greater reduction of the objective function than the one gain-based method selects (as shown in the "$\Delta_{\mathcal{O}}$" columns).



(a) Performance comparison on $\mathcal{F}_0$

(b) Performance comparison on $\mathcal{F}_1$

Figure 3. Performance comparison of McCallum's feature selection scheme and gradient-sieve on the casino gaming example for candidate pools $\mathcal{F}_0$ and $\mathcal{F}_1$.

Table 1. Comparison of time spent on each iteration (in seconds), reduction of objective function ($\Delta_{\mathcal{O}}$), prediction accuracy, and the selected features for candidate set $\mathcal{F}_0$.

	McCallum's method				gradient-sieve			
iter	time	$\Delta_{\mathcal{O}}$	accuracy (%)	feature	time	$\Delta_{\mathcal{O}}$	accuracy (%)	feature
1	32	8951.8	66.11	$T(fake, real)$	0	4317.6	66.11	$T(real, real)$
2	83	394.2	66.11	$E(real, 4)$	0	4891.5	66.11	$T(fake, fake)$
3	55	427.1	64.64	$E(real, 6)$	0	1644.0	73.92	$E(fake, 6)$
4	63	1080.5	73.89	$T(real, real)$				

Now let us take a close look at the features selected by the two methods. Recall that, for the casino gaming example, the candidate pool $\mathcal{F}_0$ has 16 features in total, including 4 transition features and 12 0^{th} order emission features.

As shown in Figure 2, the most conspicuous characteristics of the casino model are:

a) The state "real" has a higher self transition probability than "fake". In fact, according to the stationary distribution theory of Markov chain, 66% of the numbers belong to the state "real".

b) The model has a high self transition probability which means that the model tends to stay in the same state for a long time.

c) The biggest difference between the two states is that the "fake" state has a much higher probability to generate the number 6.

Given the above analysis, we can see that the gradient-based method always selects high-priority features. For $\mathcal{F}_0$, the feature subset selected includes three features $T(real, real)$, $T(fake, fake)$, and $E(fake, 6)$, where $T()$ are transition features and $E()$ are emission features. The optimal weight vector $\Lambda^\star = \{2.11, 1.48, 1.64\}$. It first selects the self transition feature $T(real, real)$ and labels all the data points as in the state "real". As a result, we get an accuracy of 66.11%. The trained weight vector also verifies this property as the self-transition feature $T(real, real)$ gets a larger weight than $T(fake, fake)$. The second important feature selected is the self-transition feature $T(fake, fake)$, which is justified by property b) listed above. The third feature selected is $E(fake, 6)$. Based on the model, if the number 6 appeared in the observation sequence, then it is likely that the underlying state should be "fake". As a result, adding this feature serves as an evidence to shift from the self transitions $T(real, real)$ to the state "fake". From the above, we see that the selected features reflect human knowledge on the model. A similar argument can be made for candidate pool $\mathcal{F}_1$.

4 Experimental Results

Based on previous comparison, we can easily see the advantage of the gradient-based feature selection scheme on CRF models, especially for tasks with large candidate pool.

We apply the gradient-based feature selection method to the leading de novo gene predictor called CONTRAST [7]. The original model includes over 33,000 features. There are four main types of features used in CONTRAST: features encoding all possible states, features encoding transitions between states, features encoding region boundaries and features encoding a state based on sequence near its po-

Table 2. Comparison of time spent on each iteration, reduction of objective function ($\Delta_{\mathcal{O}}$), prediction accuracy, and the selected features for candidate set $\mathcal{F}_1$.

	McCallum's method				gradient-sieve			
iter	time	$\Delta_{\mathcal{O}}$	accuracy (%)	feature	time	$\Delta_{\mathcal{O}}$	accuracy (%)	feature
1	199	8951.8	66.11	$T(fake, real)$	1	4317.6	66.11	$T(real, real)$
2	416	394.2	66.11	$E(real, 4)$	0	4891.5	66.11	$T(fake, fake)$
3	423	433.1	68.20	$E(fake, 66)$	0	1644.0	73.92	$E(fake, 6)$
4	1151	786.3	70.70	$T(real, real)$	1	1.3	73.92	$E(real, 3)$
5	1301	3.2	73.93	$E(real, 6)$	0	2.4	73.96	$E(fake, 66)$
6	1475	2.5	73.92	$E(fake, 24)$	0	3.2	74.10	$E(real, 52)$
7	1536	2.3	73.88	$E(fake, 52)$				
8	1559	1.1	74.00	$E(real, 33)$				

sition [7]. The sequence-based features can be further divided into three categories: features based solely on the target sequence (DNA hexamer), features based on the alignment of a particular species to the target sequence (combination of trimer) and features based on the Expressed Sequence Tag (EST) [1] evidence. It is important to note that the EST information is not always available for all genomes.

Although CONTRAST has achieved substantial success, the large set of features used in its model results in great computational demand. For a gene prediction task with around 3,000 genes in the training set, it takes around 6 hours on a server with 20 processors to finish. However, not all the 33,000 features are necessary and many of them are redundant. We apply the gradient-based method to CONTRAST and perform gene prediction on fly genomes. The performance of prediction is measured by sensitivity(*Sn*) and specificity(*Sp*) at the gene, exon and nucleotide levels. We divide the whole data set which contains around 16,000 genes evenly into four subsets.

In the following results, we are going to show that

a) The gradient-based method achieves comparable or even better gene prediction accuracy after selecting only $2\% \sim 3\%$ of the 33,000 features from the candidate pool.

b) The features selected are biologically meaningful.

c) The gradient-based method is better than a filtering approach in terms of quality.

d) The gradient-based method is applicable to real world problems with large candidate pools, for which the gain-based method is impractical due to its high computational costs.

e) The feature subset selected by the gradient-based can be directly used on statistically similar datasets and achieve comparable accuracies.

f) The gradient-based method works well even when the EST features are not available in the candidate pool.

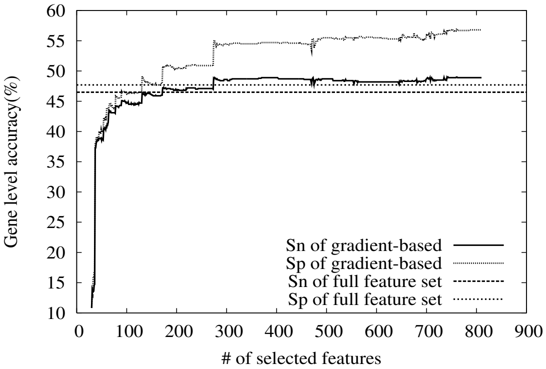
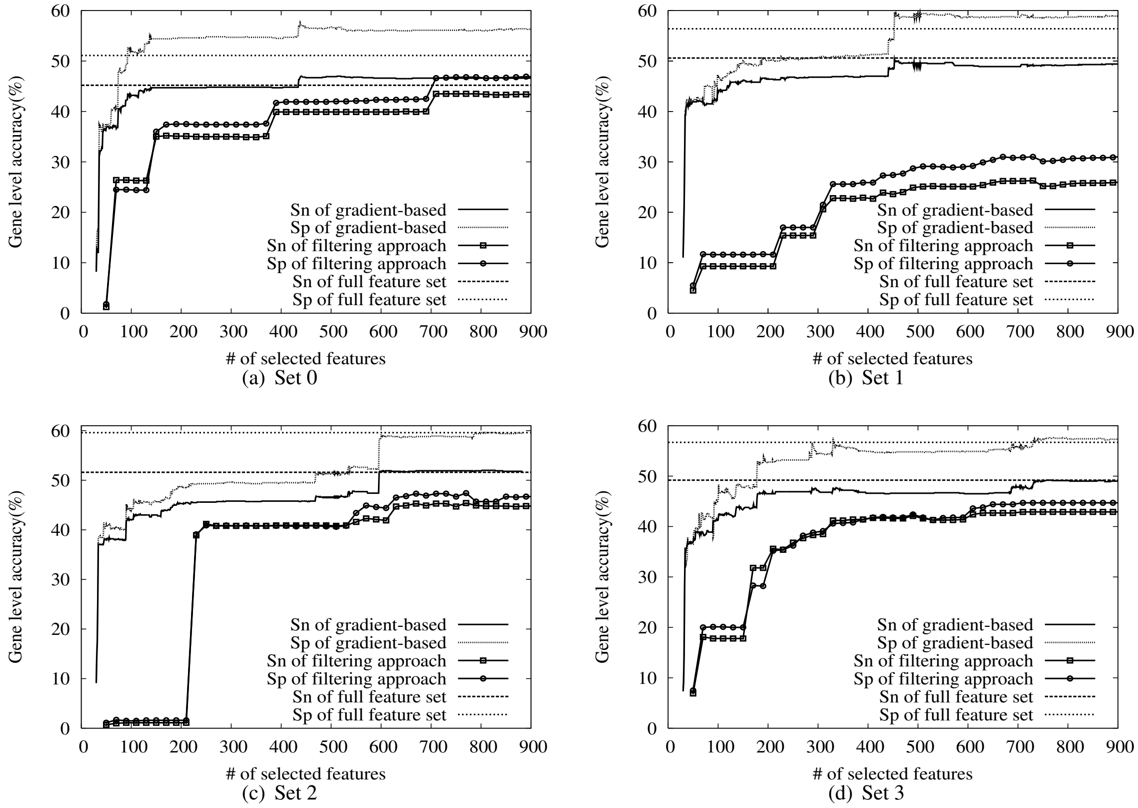


Figure 5. Performance of the gradient-based method on the gene prediction task when the candidate pool does not include EST features. The full feature set includes 32,983 features.

In Figure 4, we plot the prediction accuracies on unseen testing data for each feature selection iteration. As shown in the figure, when including these EST alignment features, the gradient-based feature selection scheme is able to outperform the original model with only around 600 features for most cases, while the original model in CONTRAST used more than 33000 features. We also apply a filtering approach [4] based on mutual information to this task. It iteratively adds features that give the maximum mutual information gain. As we can see from Figure 4, the gradient-based method gives much better accuracies than the filtering approach. Finally, we are able to confirm that the features identified by the gradient-based method are biologically meaningful. For example, all the EST features are selected in the very early stage.

As reported before, when the EST features are included in the feature set, the performance improves significantly. However, EST features are hard to obtain for many genomes. Thus, we also tested CONTRAST without the EST features and show the results in Figure 5. Due to lim-



(a) Set 0

(b) Set 1

(c) Set 2

(d) Set 3

Figure 4. Performance of the gradient-based method on the gene prediction task when the candidate pool include EST features. The full feature set includes 33,003 features.

ited space, we only present the result on one dataset, as the other datasets give similar results. As we can see, while the prediction accuracy of the original model deteriorates considerably without the EST features, the gradient-based method is able to achieve similar accuracy as the one with EST features after selecting around 800 features. Hence, it is robust even when the EST features are not available.

The running time of the gradient-based method is much less than that of the gain-based method, which is prohibitively long for this task. The gradient-based takes about 24 hours to run each set of training/testing data on the same 20-CPU cluster. Although it is longer than the 6 hours original CONTRAST uses, it gives vital information about useful features. Moreover, as we show below, the feature subset obtained from one dataset can be directly used for other datasets, giving a huge saving in future computational time.

As stated in Section 1, we expect the feature subset selected by our method to be able to generalize well to other training and testing data. Hence, we carried out another experiment to directly use the feature subset selected for one data set on the other data sets. In Table 3(a), the first two columns are the sensitivity and specificity, respectively, under the original model with 32983 features (without EST features). The following 6 columns show the prediction accuracies on set 0 by directly using the feature subsets selected using the training data set 1, set 2, and set 3, respectively. Tables 3(b), 3(c), and 3(d) are similar.

As shown in Table 3, the selected feature subsets generalize well to other training tasks. We mark in bold where we got better prediction accuracies. On average, we did 0.7% better on sensitivity, and 1.0% worse on specificity. In conclusion, the feature subset selected by our method is able to generalize well to statistically similar datasets. As a result, in the future, we can directly use the selected features instead of all the features from the candidate pool to perform similar tasks, and save computational time.

5 Conclusions

In this paper, we have presented a detailed comparison of the gain-based and gradient-based feature selection schemes for CRFs. Both methods build up a discriminative feature subset by iteratively adding the feature with the maximum potential gain estimated via different criteria.

Table 3. Performance of using the feature subset selected from one dataset on other datasets. The second row shows the number of features included, and third to fifth rows show the sensitivity and specificity at gene, exon and nucleotide levels, respectively.

(a) Performance on set 0

	CONTRAST		from set 1		from set 2		from set 3	
$\|\mathcal{S}\|$	32983		800		625		670	
Set 0	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
Gene	42.7	51.7	**42.9**	48.7	42.3	49.2	**43.4**	49.0
Exon	77.7	72.3	77.3	70.1	77.1	70.9	77.5	70.7
Nucl.	96.9	88.9	96.9	88.6	96.6	87.1	96.7	87.0

(b) Performance on set 1

	CONTRAST		from set 0		from set 2		from set 3	
$\|\mathcal{S}\|$	32983		555		625		670	
Set 1	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
Gene	46.5	47.7	**48.3**	**52.3**	**48.1**	**52.5**	**48.3**	**51.5**
Exon	76.8	65.1	**78.4**	**66.7**	**78.1**	**67.5**	**78.4**	**67.4**
Nucl.	96.5	85.7	**96.4**	85.2	96.3	85.4	**96.6**	85.3

(c) Performance on set 2

	CONTRAST		from set 0		from set 1		from set 3	
$\|\mathcal{S}\|$	32983		555		680		963	
Set 2	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
Gene	46.1	51.5	45.7	46.8	**47.0**	49.4	**46.9**	50.4
Exon	78.9	70.5	77.9	67.3	78.0	68.2	78.3	69.3
Nucl.	97.8	87.8	96.8	86.8	97.1	87.5	97.1	87.7

(d) Performance on set 3

	CONTRAST		from set 0		from set 1		from set 2	
$\|\mathcal{S}\|$	32983		555		680		625	
Set 3	Sn	Sp	Sn	Sp	Sn	Sp	Sn	Sp
Genes	46.0	52.9	45.5	49.5	**46.2**	49.6	**47.1**	49.1
Exon	78.2	70.3	77.5	68.5	77.4	68.5	77.8	67.8
Nucl.	97.5	86.4	96.5	86.1	96.9	86.2	96.5	86.4

Comparing to the gain-based method, the gradient-based method is computationally much more efficient, and effective in identifying the most promising features. We applied the gradient-based feature selection method to a state-of-the-art gene predictor CONTRAST and shown that it is able to achieve comparable or even better prediction accuracies on unseen testing data using a small fraction ($2\% \sim 3\%$) of the features in the candidate pools. Further, we have shown that features selected by the gradient-based method from a fraction of the genome dataset generalize well to the remaining of the dataset.

References

[1] M. Adams, J. Kelley, J. Gocayne, M. Dubnick, M. Polymeropoulos, H. Xiao, C. Merril, A. Wu, B. Olde, R. Moreno, and A. Et. Complementary dna sequencing: expressed sequence tags and human genome project. *Science*, 252:1651–1656, June 1991.

[2] A. Bernal, K. Crammer, A. Hatzigeorgiou, and F. Pereira. Global discriminative learning for higher-accuracy computational gene prediction. *PLoS Computational Biology*, 3:e54, 2007.

[3] C. Burge. *Identification of genes in human genomic DNA*. PhD thesis, Stanford Univerisity, 1997.

[4] H. Cheng, X. Yan, J. Han, and C.-W. Hsu. Discriminative frequent pattern analysis for effective classification. *IEEE 23rd International Conference on Data Engineering (ICDE)*, pages 716–725, 2007.

[5] Y. L. Cun, J. S. Denker, and S. A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, pages 598–605. Morgan Kaufmann, CA, 1990.

[6] S. Gross and M. Brent. Using multiple alignments to improve gene prediction. *Journal of computational biology*, 13(2):379–393, 2006.

[7] S. S. Gross, C. B. Do, M. Sirota, and S. Batzoglou. Contrast: a discriminative, phylogeny-free approach to multiple informant de novo gene prediction. *Genome Biology*, 8, 2007.

[8] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[9] K. Kira and L. A. Rendell. The feature selection problem: Traditional methods and a new algorithm. *the Tenth National Conference on Artifical Intelligence*, 1992.

[10] R. Kohavi and G. H. John. Wrappers for feature selection. *Artifical Intelligence*, 1997.

[11] I. Korf, P. Flicek, D. Duan, and M. Brent. Integrating genomic homology into gene structure prediction. *Bioinformatics*, 17:140–148, 2001.

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *International Conference on Machine Learning*, 2001.

[13] S.-I. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of markov networks using $l1$-regularization. In *Neural Information Processing Systems (NIPS 19, 2007)*.

[14] A. McCallum. Efficiently inducing features of conditional random fields. *Conference on Uncertainty in Artificial Intelligence*, 2003.

[15] S. Perkins, K. Lacker, J. Theiler, I. Guyon, and A. Elisseeff. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356, 2003.

[16] S. D. Pietra, V. D. Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:380–393, 1997.

[17] H. Stoppiglia, G. Dreyfus, R. Dubois, and Y. Oussar. Ranking a random feature for variable and feature selection. *JMRL*, 3:1399–1414, 2003.

[18] C. Sutton and A. Mccallum. *Introduction to Conditional Random Fields for Relational Learning*. MIT Press, 2006.

[19] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *the Fourteenth International Conference on Machine Learnings*.