Kernel Methods for Machine Learning
Data Challenge 2018-2019

Victor Marchais, **Louis Trezzini** and **Tong Zhao**
Team "Kernels are the new Sexy"

École des Ponts - Master MVA

{victor.marchais, louis.trezzini, tong.zhao}@eleves.enpc.fr

1 Introduction

This challenge is a binary classification task on string sequences: given a DNA sequence of 100 nucleotides, we have to predict whether this DNA region is a binding site to a specific transcription factor (TF). In this challenge, we will work with three datasets corresponding to three different TFs, each dataset containing 2 000 training examples and 1 000 testing examples. We will learn a specific classifier for each of these datasets. The evaluation metric is accuracy.

2 Algorithms

To solve this binary classification problem, we implemented both Kernel Logistic Regression and SVM (Support Vector Machine).

2.1 Kernel Logistic Regression

As suggested in class, we implemented Kernel Logistic Regression algorithm by solving iteratively a Weighted Kernel Ridge Regression problem until convergence.

2.2 Support Vector Machine

We also implemented the SVM (Support Vector Machine) algorithm by solving the dual formulation:

$$\max_{\alpha \in \mathbb{R}^n} 2\alpha^T y - \alpha^T K \alpha$$
$$\text{s. t. } \forall 1 \leq i \leq n, 0 \leq y_i \alpha_i \leq \frac{1}{2\lambda n}$$

We solve this Quadratic Programming problem using CVXOPT, a general purpose optimization library.

3 Kernels

Sequences of the DNA and RNA sequences code protein. Similar sequences are likely to produce similar proteins that have the same functions. Starting with this assumption we implemented the following kernels:

3.1 Spectrum Kernel

Spectrum kernel [Leslie *et al.*, 2001] is a simple yet efficient string kernel which build a binary feature map where non-zero values correspond to all sub-sequences of length k of a given sequence. Since the alphabet is of size n (which in our case is 4 for nucleotides `A`, `T`, `C`, `G`), the feature vector is of dimension n^k. This quantity grows quickly with k, but the feature vector is sparse: for a given sequence, there are at most $|s| - k + 1$ non-zero coefficients. This allows for efficient approaches to compute kernel values. Thus, we implemented this kernel using a Trie (prefix tree).

$$\Phi_s^{\text{spe}}(S) = \begin{cases} 1 & \text{if } s \in S \\ 0 & \text{otherwise} \end{cases}$$

$$K^{\text{spec}}(S_1, S_2) = \sum_{s:|s|=k} \Phi_s^{\text{spec}}(S_1)\Phi_s^{\text{spec}}(S_2) \tag{1}$$

3.2 Weighted Spectrum Kernel

We thought it would be interesting to consider not only all sub-sequences of length k, but rather all sub-sequences of length between 4 and k (for $k \geq 4$), and to affect different weights to different lengths of sub-sequences.

We chose a geometric decrease, with a weight w:

$$K^{w-\text{spec}}(S_1, S_2) = \sum_{s:4\leq|s|\leq k} w^{k-|s|}\Phi_s^{\text{spec}}(S_1)\Phi_s^{\text{spec}}(S_2) \tag{2}$$

3.3 Kernels for mutations

Sequences are not immutable, mutations can occur. Three transformations are possible: insertion, deletion, and mutation. To take into account these possible transformation we implemented two more kernels: Mismatch [Eskin *et al.*, 2003] and Substring.

Mismatch Kernel

For each word s of length k from the alphabet, we consider $\Phi_s^{\text{mis}}(S) = w^{m(s,S)}$ with w a chosen weight and $m(s, S)$ the minimum number such that there exist $s_0 \in S$ that can give the sequence s after $m(s, S)$ mutations. Obviously, if $s \in S$, $\Phi_s^{\text{mis}}(S) = 1$. To avoid exponential computation, we only consider at most 1 mismatch:

$$\Phi_s^{\text{mis}}(S) = \begin{cases} 1 & \text{if } s \in S \\ w & \text{if } s \in S \text{ after one mutation} \\ 0 & \text{otherwise} \end{cases}$$

$$K^{\text{mis}}(S_1, S_2) = \sum_{s:|s|=k} \Phi_s^{\text{mis}}(S_1)\Phi_s^{\text{mis}}(S_2) \qquad (3)$$

Substring Kernel

For each word s of length k from the alphabet, we consider $\Phi_s^{\text{sub}}(S) = w^{j(s,S)}$ with w a chosen weight and $j(s, S)$ the minimum number such that there exist $s_0 \in S$ that can give the sequence s after inserting $j(s, S)$ characters in the sequence. Obviously, if $s \in S$, $\Phi_s^{\text{sub}}(S) = 1$. To avoid exponential computation, we only consider at most j jumps.

$$K^{\text{sub}}(S_1, S_2) = \sum_{s:|s|=k} \Psi_s^{\text{sub}}(S_1)\Psi_s^{\text{sub}}(S_2) \qquad (4)$$

3.4 Normalized Kernels

In practice, we noticed that all kernels above worked better after normalization:

$$\tilde{K}(S_1, S_2) = \frac{K(S_1, S_2)}{\sqrt{K(S_1, S_1)K(S_2, S_2)}} \qquad (5)$$

4 Methodology

We used 5-folds cross-validation to select all the hyper-parameters, then we retrained the classifier on the whole training set before making our predictions on the test set.

For all of our early experiments, SVM seemed to outperform Kernel Logistic Regression. We decided to only use SVM for subsequent experiments.

We first tried all individual kernels with various parameters, then we tried to sum the best ones we had.

For every kernel (or sum of kernels) and every dataset, we used the library `hyperopt` to find the best regularization parameter λ. This parameter plays an essential role in the performance of the classifier, because very few regularization would lead us to overfit the data, but too much regularization would prevent us from learning altogether. Here, regularization is particularly important to prevent overfitting, because the kernels we designed have very high dimensional feature vectors (hundreds of thousands of coefficients) and λ allows us to trade-off between training accuracy and smoothness (hopefully validation accuracy).

5 Best submission

Our best submission is using an SVM classifier, whose kernel is the sum of all the **normalized** kernels below:

Kernel	Set 1	Set 2	Set 3
Spectrum	Not used	Not used	$k \in \{12, 15\}$
Weighted Spectrum	$k = 12$	$k = 10$	$k = 10$
	$w = 0.75$	$w = 0.75$	$w = 0.5$
Mismatch	$k = 8$	$k = 10$	Not used
	$m = 1$		
	$w = 0.75$	$w = 0.5$	
Substring	$k = 8$	$k = 12$	$k = 12$
	$j = 2$		
	$w = 0.75$	$w = 0.75$	$w = 0.5$

This submission scores as follows:

Evaluation	Accuracy	Ranking
Local Cross-Validation	0.70350	-
Public Leaderboard	0.71666	16
Private Leaderboard	0.71066	2

This was our absolute best submission using local cross-validation score, public and private leaderboard score. This shows we are not overfitting the data.

6 Conclusion

By using the knowledge learned in the class, we achieved to build a model predicting if a DNA region is a binding site to a given TF. Our final model is a SVM classifier based on a sum kernel of four widely used string kernels, with hyper-parameters selected by cross-validation strategy. It is shown that the model achieved a good score on the whole test dataset without over-fitting. With more time, we would have liked to explore using a weighted sum of kernels, by implementing Multiple Kernel Learning (MKL).

References

[Eskin *et al.*, 2003] Eleazar Eskin, Jason Weston, William S Noble, and Christina S Leslie. Mismatch string kernels for svm protein classification. In *Advances in neural information processing systems*, pages 1441–1448, 2003.

[Leslie *et al.*, 2001] Christina Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for svm protein classification. In *Biocomputing 2002*, pages 564–575. World Scientific, 2001.