Kernel Methods for Machine Learning Final Project
Team name : KERNENS

Hurault Samuel
Ecole Normale Superieure Paris-Saclay
samuel.hurault@ens-paris-saclay.fr

Matthieu Cordonnier
Ecole Normale Superieure Paris-Saclay
matthieu.cordonnier@ens-paris-saclay.fr

Abstract

The goal of the data challenge is to develop a Kernel classification method predicting whether a DNA sequence region is binding site to a specific transcription factor. We work with three datasets corresponding to three different TFs.

1. Introduction

In all our work, we work with the SVM Kernel classification algorithm. Once SVM coded, we developed and experiences different kind of kernels studied in class. We also added classical data preprocessing and generalization methods. All the parameters involved will be tuned by cross-validation with validation sets made of 20% of the original data.

2. Linear, Polynomial and Gaussian Kernels

2.1. Given input embedding

We first developed the classical SVM and 2-SVM algorithms. They involve a parameter C. We first use the proposed embedding as inputs of the algorithm. We propose three different kernels : the linear kernel, the polynomial kernel (with parameter d) and the Gaussian kernel (with parameter σ). The training data (the embedding matrices) is normalized before SVM computation. We tune the following hyper-parameters by successive cross-validations :

- C for the linear kernel
- C, d for the polynomial kernel
- C, σ for the gaussian kernel

We test the C values in $\{2^k\}_{k=-9,..,7}$, d in $\{1, 2, 3, 4\}$ and σ in $\{10^k\}_{k=-4,..,4}$.

Type	Kernel	Optimal parameters	Accuracy
1	Linear kernel	$C = 0.0625$	0.575
1	Polynomial kernel	$C = 4$, $d = 1$	0.5675
1	Gaussian kernel	$C = 32$, $sig = 100$	0.6
2	Linear kernel	$C = 0.015625$	0.655
2	Polynomial kernel	$C = 0.0078$, $d = 2$	0.655
2	Gaussian kernel	$C = 32$, $sig = 10$	0.685
3	Linear kernel	$C = 0.0078$	0.625
3	Polynomial kernel	$C = 0.03125$, $d = 1$	0.6275
‘ 3	Gaussian kernel	$C = 1$, $sig = 1$	0.645

Table 1. Parameter tuning for SVM with given input embedding

2.2. dna2vec pretrained embedding

A new DNA sequence embedding method can be inspired from techniques developed in Natural Language Processing. In [2], the author train an embedding dna2vec based on the popular word embedding model word2vec. They obtain a representations of variable-length k-mers (sub-sequence of size k). We don't have enough data to train our own dna2vec embedding but we can try to use the one given in by the authors [1]. Such experiment did not improve the accuracy.

3. Spectrum and Mismatch Kernels

We compute the Spectrum and Mismatch kernels studied in class. They are linear kernels applied on a particular feature space. For the spectrum kernel, the feature is the number of occurrences of sub-sequence u of length n. For the mismatch kernel, it is the number of occurrences of u in up to m mismatches. For the spectrum kernel, we tune the parameters C and n. We search n in $3, .., 8$. For the mismatch kernel, we only consider $m = 1$ because of limitation in running time. We observe that the performances of this kernel are not as good as the ones obtained with the spectrum kernel.

[1] https://github.com/pnpnpn/dna2vec

Type	Kernel	Optimal parameters	Accuracy
1	Spectrum kernel	$C = 2^{-7}, n = 6$	0.618
2	Spectrum kernel	$C = 2^{-7}, n = 7$	0.73
3	Spectrum kernel	$C = 2^{-7}, n = 7$	0.642

Table 2. Parameter tuning for SVM with Spectrum and Mismatch kernels

4. Data augmentation

We observe that the SVM is strongly overfitting (accuracy of 1. on the training data). To reduce overfitting, we try to add noisy in the training data : for every sequence, we randomly select $p\%$ of the sequence and create a new sequence where each selected element is randomly selected between the four possible DNA letters. Such a method did not improve the accuracy results.

5. Bagging

To improve the classification performance of our SVM model, following [1], we propose to use the SVM with bagging (bootstrap aggregating). Each individual SVM is trained independently using randomly chosen training samples. Then, they are aggregated into to make a collective decision. Their intuition for such a method is that sometimes, the support vectors obtained from the learning is not sufficient to classify all unknown test examples completely. So, we can not guarantee that a single SVM always provides the global optimal classification performance over all test examples. We randomly sample N different training sets with 90% of the training data, without replacement. The final prediction is obtained with a voting scheme : we choose the class that obtained the maximum of votes from each of the N models. We tune the parameter N in $\{3, 5, 7, 10\}$ by cross-validation : We limit N to 10 due to computational time limits.

Type	best N value	Accuracy
1	10	0.62
2	7	0.735
3	10	0.645

Table 3. Parameter tuning of the number of models N in the bagging method

6. Conclusion

In this word we developed and optimized a SVM - based binary classification model for ADN sequences. Our best model is composed of a SVM with a spectrum kernel. We tuned the hyper-parameters for each type of data independently. With more time, we would have tried to combine different kernels thanks to Multiple Kernel Learning. We finally achieved an accuracy of 0.66600 on the private evaluation set.

References

[1] H.-C. Kim, S. Pang, H.-M. Je, D. Kim, and S.-Y. Bang. Support vector machine ensemble with bagging. In S.-W. Lee and A. Verri, editors, *Pattern Recognition with Support Vector Machines*, pages 397–408, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[2] P. Ng. dna2vec: Consistent vector representations of variable-length k-mers. *arXiv e-prints*, page arXiv:1701.06279, Jan 2017.