DNA Sequence Classication with Kernel Methods

Pierre Fernandez
pierre.fernandez@polytechnique.edu

Paul Jacob
paul.jacob@polytechnique.edu

Clément Nguyen
clement.nguyen@eleves.enpc.fr

Team: Fernandez-Jacob-Nguyen - **Public score:** 0.67866 (6th) - **Private score:** 0.68333 (1st)

1 Introduction

This is our report for the Kaggle data challenge done in the scope of the course "Machine Learning with Kernel Methods" by Jean-Philippe Vert and Julien Mairal, for the MVA master program (2020-2021). The goal of this challenge is to implement kernel-based classification methods to predict whether a DNA sequence region is binding site to a specific transcription factor. Here we present the methods we used, our experiments and some results. The best submissions were obtained using different Mismatch Kernels applied directly on the DNA sequences, combined as one sum Kernel, and then used into a SVM as binary classifier. We give further details about our experiments in the following sections. Our code is available on Github[1].

2 Task & Datasets

The predictions are made over 3 datasets. For each of these datasets, we have 2000 labeled training sequences of 101 nucleotides, as well as 1000 unlabeled test sequences that we want to classify. In order to measure the quality of a model before submitting, we first performed training-validation splits on the labeled datasets, with 0.8/0.2 ratio. This method is very fast to implement and allows to grasp faster if a given Kernel will give good accuracy. However, it leads to only 400 samples in the validation set, which is prone to having significant variance. Hence, for more accurate validation scores, we decided to use k-fold cross validation (with $k = 5$) on the labeled datasets, and used the mean validation accuracy as our evaluation criterion.

3 Kernels & Classifiers

Kernels that operate on vectors Along with the sequences of nucleotides provided in the data challenge, there are vectorized versions given as matrices, for each dataset. The first Kernels that we implemented dealt with this format. Among them, the linear Kernel, the polynomial Kernel and the gaussian Kernel.

Kernels that operate on string sequences Applying Kernels especially designed for biological sequences allowed to capture more information since the vectorization step loses information. We decided to focus on two types of Kernels on sequences: the Spectrum and Mismatch Kernels.

The Spectrum Kernel [1] is a sequence-similarity Kernel, designed for the protein classification problem. It consists in counting the occurrences $\Phi_u(x)$ of each given k-mer u in the sequence x, then the k-spectrum Kernel reads $K(x, y) = \sum_u \Phi_u(x)\Phi_u(y)$.

[1] https://github.com/pierrefernandezgenestier/DNA-sequence-classification-with-Kernel-methods

For the Mismatch Kernel [2], instead of simply counting the occurrences of the k-mer u in the sequence x, we allow m mismatches. It is more realistic than the Spectrum Kernel since we can imagine that the TF can bound to a DNA fragment even if 1 or 2 nucleotides do not match. To compute the Mismatch Kernel, for each k-mer u appearing in a sequence of the dataset, we first pre-computed the "Mismatch neighborhood" around u, i.e. the set of k-mers differing from u by at most m mismatches. Using the pre-computed neighborhoods, it is easy to build the feature map $\Phi(x) = [\Phi_u(x)]_u$, given an input sequence x. To allow fast computation, we stored the feature maps in sparse matrices and used sparse matrix multiplication. As proposed in [2], we found useful normalizing the Kernel, such that for two sequences x and y, $K(x, y) = \langle\Phi(x), \Phi(y)\rangle/(||\Phi(x)|| \cdot ||\Phi(y)||)$.

The Weighted Sum Kernel The weighed sum Kernel of $K_1, ..., K_n$ with weights $w_1, ..., w_n$ consists in computing a new Kernel $K = \sum_{i=1}^n w_i K_i$. Summing Kernels this way amounts to concatenating their feature space representations and allows to retrieve information from these different feature spaces, as well as avoids overfitting by getting information from different sources. We used this type of Kernel mainly to combine Mismatch Kernels obtained with different values of k and m, hoping it would allow to retrieve structural information of the proteins at different scales.

Classifiers As advised, we first implemented the Kernel Ridge Regression (KRR) and the Kernel Logistic Regression (KLR). KRR solves $\text{argmin}_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n (y_i - f(x_i))^2 + C\|f\|_{\mathcal{H}}^2$, while the KLR solves $\text{argmin}_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n \ln(1 + e^{-y_i f(x_i)}) + C\|f\|_{\mathcal{H}}^2$. Then we moved on to large margin classifiers for the Hinge loss, i.e. the SVM that solves $\text{argmin}_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n \max(1 - y_i f(x_i), 0) + C\|f\|_{\mathcal{H}}^2$.

All the classifiers were implemented using Part 3 of the course's slides, with no other library than `cvxopt`, `scipy` and `numpy`.

4 Results

A SVM with a Gaussian Kernel ($C = 10, \gamma = 10$) already gives a good baseline as we obtained 0.605 on the public dataset. However, most of our experiments on the vectorized data could not do better than 0.62 when cross-validating. Hence, we quickly decided to move to experiments on the raw data with the Spectrum and Mismatch Kernels. For instance, a SVM with $C = 1$ and Mismatch Kernel with $k = 12, m = 2$ alone gave 0.679 on the public dataset and 0.672 on the private one. Our best submission consisted in the sum of many Mismatch Kernels, namely $(k, m) \in \{(18, 3), (15, 3), (13, 2), (12, 2), (10, 1), (8, 1), (5, 1)\}$. It led to 0.683 on the private leaderboard and 0.677 on the public one. The Mismatch Kernels that were chosen in this sum were the ones that, alone, obtained good scores on the cross-validation steps (see table 1 for the cross-validation results of the Mismatch Kernels). It can be noted that the scores obtained during the cross-validation steps were higher than the ones on Kaggle, which is not very surprising given the small size of the datasets.

5 Conclusion

In this challenge, we implemented Kernel methods from scratch, and obtained promising results on the DNA classification task with 0.683 overall accuracy (and a first place in the private leaderboard!). This score could be improved by doing more hyperparameter tuning. For instance, we have not optimized the weights of the sum Kernel and the choice of C in the SVM as much as we could have, since it was a bit computationally expansive.

Appendix

Kernel	k=5,m=1	k=8,m=1	k=10,m=1	k=12,m=2	k=13,m=2	k=15,m=3	k=18,m=3
Data 0	0.619	0.656	0.661	0.679	0.673	0.675	0.644
Data 1	0.620	0.652	0.676	0.666	0.654	0.665	0.603
Data 2	0.719	0.747	0.764	0.750	0.742	0.756	0.700

Table 1: Mean cross-validation accuracy of Mismatch Kernels over the three datasets (SVM, C=1).

References

[1] C. Leslie, E. Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for svm protein classification. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 564–75, 2002.

[2] Christina S. Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston, and William Stafford Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4):467–476, 01 2004.