DATA CHALLENGE - KERNEL METHODS (2024-2025)

Lucas Versini
lucas.versini@polytechnique.edu / lucas.versini01@gmail.com

Team: Lucas Versini / Public score: 71.4% (4^{th}) / Private score: 71.6% (4^{th})

1 Introduction

This report documents my work for the Kaggle challenge in the *Kernel Methods for Machine Learning* course, as part of the MVA Master's program.

The challenge involves implementing kernel methods to predict whether a given DNA sequence corresponds to a binding site for a specific transcription factor.

All our code can be found on GitHub[1].

2 Task description and dataset

We were provided with three datasets, each containing 2,000 labeled sequences and 1,000 unlabeled sequences. Each sequence is composed of 101 nucleotides, represented as A, G, T, or C. The sequences are either binding (label 1) or non-binding (label 0).

For each dataset, we are given both the raw string sequences and a feature matrix derived using a bag-of-words representation.

For our experiments, we performed 5-fold cross-validation to ensure sufficient training and validation samples for model evaluation. The models were selected based on their mean validation accuracy.

3 Classifiers

The labels were converted to -1 and +1 (i.e., label 0 becomes -1 and label 1 becomes +1) for compatibility with the classifiers.

We used the following classifiers, which were introduced in the course:

- **Ridge regression:** Solves
$$\arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\|f\|_{\mathcal{H}}^2.$$
- **Support Vector Machines (SVM):** Solves
$$\arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}\max(0, 1 - y_i f(x_i)) + C\|f\|_{\mathcal{H}}^2.$$
- **Logistic regression:** Solves
$$\arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}\ln\left(1 + e^{-y_i f(x_i)}\right) + C\|f\|_{\mathcal{H}}^2.$$

The hyperparameters (λ or C) for each model were tuned through grid search, and the hyperparameter yielding the best mean validation accuracy was selected. We implemented the classifiers using `numpy` and `cvxopt` (for solving quadratic problems in SVM).

4 Kernels

We implemented several kernel functions, some applied to the feature matrices and others directly to the raw string sequences.

4.1 Matrix kernels

We used classical kernels on the feature matrices, including linear, polynomial, and Gaussian kernels.

The best performance, shown in Table 1, was achieved with a Gaussian kernel combined with an SVM classifier. The optimal hyperparameters (C, σ) for the three datasets were respectively $(C, \sigma) = (3.0, 0.05)$, $(C, \sigma) = (9.0, 0.05)$ and $(C, \sigma) = (0.9, 0.05)$.

We then explored kernels that operate directly on the raw DNA string sequences, as these contain more detailed information than the bag-of-words matrices.

[1] https://github.com/lucas-versini/Kernel-Methods

	Dataset 1	Dataset 2	Dataset 3	Kaggle (public)	Kaggle (private)
Accuracy	57.1%	69.3%	65.3%	66.3%	65.9%

Table 1: Results of Gaussian kernel with SVM classifier (5-fold cross-validation and Kaggle scores).

4.2 String kernels

We mostly used two types of string kernels: the spectrum kernel [1] and the mismatch kernel [2]. We also tried using other kernels, such as edit distance-based kernels [3] or Fisher kernel [4], but they did not perform as well, and we did not investigate these any further.

Since the spectrum kernel is a special case of the mismatch kernel, we will focus on the latter.

The mismatch kernel is defined as follows. For integers $k \geq 1$ and $m \geq 0$, and for a finite alphabet $\mathcal{A}$ (in our case, $\mathcal{A} = \{A, C, G, T\}$), we consider the set of k-mers $\mathcal{A}^k$. For a sequence $x \in \mathcal{A}^n$ (with $n = 101$), we embed x into a vector $\Phi_{k,m}(x) \in \mathbb{N}^{\mathcal{A}^k}$, where each element counts the occurrences of k-mers in x that differ from a given k-mer $\alpha \in \mathcal{A}^k$ by at most m mismatches. The (k, m)-mismatch kernel is then given by $K_{k,m}(x, y) = \langle \Phi_{k,m}(x), \Phi_{k,m}(y) \rangle$.

The k-spectrum kernel is simply the $(k, 0)$-mismatch kernel.

To compute the mismatch kernel, we used a method that may not be the most efficient, but which worked well here, and which consists in precomputing the relevant k-mers, as well as the k-mers that are equal up to m mismatches.

4.3 Sum of kernels

Inspired by ensemble methods, we combined several kernels by summing them. The sum of positive definite kernels remains positive definite, which allowed us to experiment with combinations.

We tested various values for k and m and optimized the classifier's hyperparameters. The combination of kernels that yielded the best 5-fold cross-validation score was:

$$(k, m) \in \{(5, 1), (8, 1), (10, 1), (12, 1), (13, 1), (15, 1)\},$$

with corresponding weights: $0.19, 0.17, 0.22, 0.08, 0.10, 0.24$. The results are shown in Table 2.

We also tried using different kernels for the different datasets, but found no combination that was better than the one above.

We also use normalization as described in [2], by considering $K'(x, y) := \dfrac{K(x, y)}{\sqrt{K(x, x)K(y, y)}}$.

	Dataset 1	Dataset 2	Dataset 3	Kaggle (public)	Kaggle (private)
Accuracy	62.9%	77.5%	65.7%	71.3%	71.6%

Table 2: Results of summing mismatch kernels (5-fold cross-validation and Kaggle scores).

Adding a Gaussian kernel on top of these mismatch kernels yielded a small improvement for the 5-fold cross-validation and for the Kaggle public score (71.4 %), but none for the Kaggle private score.

5 Conclusion

In this challenge, we implemented some kernel methods from scratch, and obtained interesting results on a DNA classification task.

Many improvements are still possible, among which the use of more sophisticated kernels, the tuning of some hyperparameters, etc.

References

[1] Christina Leslie, Eleazar Eskin and William Stafford Noble. The spectrum kernel: a string kernel for SVM protein classification. Pac Symp Biocomput. 2002:564-75. PMID: 11928508.

[2] Christina S. Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston and William Stafford Noble. Mismatch string kernels for discriminative protein classification. Bioinformatics, Volume 20, Issue 4, March 2004, Pages 467–476.

[3] Michel Neuhaus and Horst Bunke. Edit distance-based kernel functions for structural pattern classification. Pattern Recognition, Volume 39, Issue 10, 2006, Pages 1852-1863.

[4] T. Jaakkola, M. Diekhans and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. Proc Int Conf Intell Syst Mol Biol. 1999:149-58. PMID: 10786297.