A mutual information kernel for sequences

Marco Cuturi
Computational Biology Group
Ecole des Mines de Paris
35 rue Saint Honoré
77300 Fontainebleau
marco.cuturi@ensmp.fr

Jean-Philippe Vert
Computational Biology Group
Ecole des Mines de Paris
35 rue Saint Honoré
77300 Fontainebleau
jean-philippe.vert@ensmp.fr

***Abstract*— We propose a new kernel for strings which borrows ideas and techniques from information theory and data compression. This kernel can be used in combination with any kernel method, in particular Support Vector Machines for protein classification. By incorporating prior assumptions on the properties of the alphabet and using a Bayesian averaging framework, we compute the value of this kernel in linear time and space, benefiting from previous achievements proposed in the field of universal coding. Encouraging classification results are reported on a standard protein homology detection experiment.**

I. Introduction

The need for efficient analysis and classification tools for sequences is more than ever a core problem in most application fields of statistical learning such as computational biology. In particular, the availability of an ever-increasing quantity of biological sequences calls for efficient and computationally feasible algorithms to detect functional similarities between DNA or amino-acid sequences, cluster them, and annotate them.

Recent years have witnessed the rapid development of a class of algorithms called *kernel methods* [20] that may offer useful tools for these tasks. In particular, the Support Vector Machine (SVM) algorithms [4], [24] provide state-of-the-art performance in many real-world problems of classifying objects into predefined classes. SVMs have already been applied with success to a number of issues in computational biology, including but not limited to protein homology detection [13], [16], [19], [2], [26] functional classification of genes [17], [25], or prediction of gene localization [11]. A more complete survey of the application of kernel methods in computational biology is presented in the forthcoming book [21].

The basic ingredient shared by all kernel methods is the kernel function, that measures similarities between pairs of objects to be analyzed or classified. While early-days SVM focused on the classification of vector-valued objects, for which kernels are well understood, recent attempts to use SVM for the classification of more general objects have resulted in the development of several kernels for strings [27], [10], [13], [15], [16], [19], [2], [26], graphs [14], or even phylogenetic profiles [25].

A useful kernel for protein sequences should have several properties. It should be rapid to compute (typically, have a linear complexity with respect to the lengths of the compared sequences), represent a biologically relevant measure of similarity, be general enough to be applied without tuning on different datasets, yet efficient in terms of classification accuracy. Such an ideal kernel probably does not exist, and different kernels might be useful in different situations. For large-scale or on-line applications, the computation cost becomes critical and only fast kernels, such as the spectrum [15] and mismatch [16] kernels can be accepted. In applications where accuracy is more important than speed, slower kernels that include more biological knowledge, such as the Fisher [13] or local alignment [26] kernels might be accepted if they improve the performance of a classifier.

Our contribution in this paper is to introduce a new class of kernels for strings that are both rapid to compute (they have a linear-time complexity in time and memory), while still including biological knowledge. The biological knowledge takes the form of a family of probabilistic models for sequences supposed to be useful to model general classes of proteins. The ones we consider are variable-length Markov chains, also known as context-tree models [28] or probabilistic suffix trees [1]. These models offer three advantages: first, they have been shown to be useful to represent protein families [1], [9], second, they can have different degrees of generality by varying the suffix-tree, allowing then to model larger or smaller classes of sequences, and third, their structure enables us to derive a kernel that can be implemented in linear time and space with respect to the sequence length. The last two features would not be shared by more complex models such as hidden Markov models [8]. A second source of biological information is represented by a prior distribution on the models, including the use of Dirichlet mixtures [8] to take into account similarities between amino-acids.

As opposed to the classical use of probabilistic models to model families of sequences [1], [9] or to the Fisher kernel, we do not perform any parameter or model estimation. We rather project each sequence to be compared to the set of all distributions in the probabilistic models, and compare different sequences through their respective projections. The resulting kernel belongs to the class of mutual information kernels introduced in [23]. Formally, the computation of the kernel boils down to computing some posterior distribution for pairs of sequences in a Bayesian framework. The computation can be performed efficiently thanks to a clever factorization of the family of context-tree models using a trick presented in [28].

The resulting kernel can be interpreted in the light of noiseless coding theory [7]: it is related to the gain in redundancy when the two sequences compared are compressed together, and not independently from one another.

The paper is organized as follows. In Section II we present the general strategy of making mutual information kernels from families of probabilistic models. In Section III we define a kernel for protein sequences based on context-tree models. Its efficient implementation is presented in Section IV, before proposing a redundancy interpretation of its value in section V. Finally, experimental results on a benchmark problem of remote homology detection are presented in Section VI

II. Probabilistic models and mutual information kernels

A (parametric) probabilistic model on a measurable space $\mathcal{X}$ is a family of distributions $\{P_\theta, \theta \in \Theta\}$ on $\mathcal{X}$, where θ is the parameter of the distribution P_θ. Typically, the set of parameters Θ is a subset of $\mathbb{R}^n$, in which case n is called the dimension of the model. As an example, a hidden Markov model (HMM) for sequences is a parametric model, the parameters being the transition and emission probabilities [8]. A family of probabilistic models is a family $\{P_{f,\theta_f}, f \in \mathcal{F}, \theta_f \in \Theta_f\}$, where $\mathcal{F}$ is a finite or countable set, and $\Theta_f \subset \mathbb{R}^{\dim(f)}$ for each $f \in \mathcal{F}$, where $\dim(f)$ denotes the dimension of f. An example of such a family would be a set of HMMs with different architectures and numbers of states. Probabilistic models are typically used to model sets of elements $X_1, \ldots, X_n \in \mathcal{X}$, by selecting a model $\hat{f}$ and a choosing a parameter $\hat{\theta}_{\hat{f}}$ that best "fits" the dataset, using criteria such as penalized maximum likelihood or maximum a posteriori probability [8].

Alternatively, probabilistic models can also be used to characterize each single element $X \in \mathcal{X}$ by the representation $\phi(X) = \left(P_{f,\theta_f}(X)\right)_{f\in\mathcal{F},\theta_f\in\Theta_f}$. If the probabilistic models are designed in such a way that each distribution is roughly characteristic of a class of objects of interest, then the representation $\phi(X)$ quantifies how X fits each class. In this representation, each distribution can be seen as a filter that extracts from X an information, namely the likelihood of X under this distribution, or equivalently how much X fits the class modelled by this distribution.

Kernels are real-valued function $\mathcal{K} : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ that can be written in the form of a dot product $\mathcal{K}(X,Y) = \langle\psi(X), \psi(Y)\rangle$ for some mapping ψ from $\mathcal{X}$ to a Hilbert space [20]. Given the preceding mapping ϕ, a natural way to derive a kernel from a family of probabilistic models is to endow the set of representations $\phi(X)$ with a dot product, and set $\mathcal{K}(X,Y) = \langle\phi(X), \phi(Y)\rangle$. This can be done for example if a prior density $\pi(f, d\theta_f)$ can be defined on the set of distributions in the models, by considering the following dot product:

$$\begin{aligned}\mathcal{K}(X,Y) &= \langle\phi(X), \phi(Y)\rangle \\ &\stackrel{\text{def}}{=} \sum_{f\in\mathcal{F}} \pi(f) \int_{\Theta_f} P_{f,\theta_f}(X) P_{f,\theta_f}(Y) \pi(d\theta_f|f). \quad (1)\end{aligned}$$

By construction, the kernel (1) is a valid kernel, that belongs to the class of mutual information (MI) kernels [23]. Observe that contrary to the Fisher kernel that also uses probabilistic models to define kernel, no model or parameter estimation is required in (1). Intuitively, for any two elements X and Y the kernel (1) automatically detects the models and parameters that explain both X and Y well.

There is of course some arbitrary in this kernel, both in the definition of the models and in the choice of the prior distribution π. This arbitrary can be used to include prior (biological) knowledge in the kernel. For example, if one wants to detect similarity with respect to families of sequences known to be adequately modelled by HMMs, then using HMM models constrains the kernel to detect such similarities. We use this idea below to define a set of models and prior distributions for protein sequences.

As the likelihood of a sequence under the models we define below decreases roughly exponentially with its lengths, the value of the kernel (1) can be strongly biased by differences in length between the sequences, and can take exponentially small values. This is a classical issue with many string kernels that leads to bad performance in classification with SVM [22], [26]. This undesirable effect can easily be controlled in our case by normalizing the likelihoods as follows:

$$\mathcal{K}_\sigma(X,Y) = \sum_{f\in\mathcal{F}} \pi(f) \int_{\Theta_f} P_{f,\theta_f}(X)^{\frac{\sigma}{|X|}} P_{f,\theta_f}(Y)^{\frac{\sigma}{|Y|}} \pi(d\theta_f|f). \quad (2)$$

where σ is a width parameter and $|X|$ and $|Y|$ stand for the lengths of both sequences. Equation (2) is clearly a valid kernel (only the feature extractor ϕ is modified), and the parameter σ controls the range of values it takes.

III. A mutual information kernel based on context-tree models

In this Section we derive explicitly a MI kernel for strings based on context-tree models with mixture of Dirichlet priors. Context-tree models are Markovian models which define an efficient framework to describe constraints on amino-acid successions in proteins, as validated by their use in [1], [9]. Dirichlet priors offer a biologically meaningful estimation of the likelihood of such transitions by giving an a-priori knowledge on the multinomial parameters which parameterize Markovian models transitions.

A. Framework and notations

Starting with basic notations and definitions, let E a finite set of size d called the alphabet. Practically speaking E can be thought of the 20 letters alphabet of amino-acids. For a given depth $D \in \mathbb{N}$ corresponding to the maximal memory of our Markovian models we note M the set of strings of E shorter than D, i.e. $M = \cup_{i=0}^{D} E^i$. We define $\mathcal{X} = \cup_{n=0}^{\infty} (E^{D+1})^n$ the set on which we define our kernel. Observe that we do not define directly the kernel on the set of finite-length sequences, but rather in a slightly more general framework where we focus on lists of transitions. We thus transform sequences into

finite lists of $D+1$ grams, which can each be divided into a $context$ (i.e a D-long subsequence of the initial sequence) and the $letter$ which is next to it. This transformation is justified by the fact that we will always consider Markovian models of maximal depths D below. An element $X \in \mathcal{X}$ can therefore be written as $X = \{x^i = x_c^i x_l^i\}_{i=1..N_X}$ where N_X is the cardinal of X and for all i, $x^i \in E^{D+1}$ can be divided into a context $x_c^i \in E^D$ and an output letter x_l^i. We also note $\varnothing$ the empty word.

Note that the set $\mathcal{X}$ endowed with a list concatenation operation, denoted as '+', is an abelian semigroup with identical involution (see [3]). The kernel which we propose in this paper can be considered as a semigroup kernel (setting aside renormalization on lengths which we use for practical purposes) on $\mathcal{X}$, a viewpoint which could make our approach the only valid one to define a kernel on $\mathcal{X}$ as a function of the merger of two lists of transitions, namely of the form $\mathcal{K}(X,Y) = \varphi(X+Y)$. Indeed, the Bochner theorem proposed by [3] in the case of abelian semigroups states that any exponentially bounded kernel admits an integral representation of semi-characters on $\mathcal{X}$. This structure fits precisely the additive bayesian mixture framework of MI kernels which we use below.

B. Context-tree models

Context-tree distributions require the definition of a complete suffix dictionary (c.s.d) $\mathcal{D}$, a c.s.d being a finite set of words of $M\backslash\{\varnothing\}$ such that any left-infinite sequence has a suffix in $\mathcal{D}$, but no word in $\mathcal{D}$ has a suffix in $\mathcal{D}$. We note $L(\mathcal{D})$ the length of the longest word contained in $\mathcal{D}$ and $\mathcal{F}_D$ the set of c.s.d $\mathcal{D}$ that satisfy $L(\mathcal{D}) \leq D$. Once this tree structure is set, we can define a distribution on $\mathcal{X}$ by attaching one multinomial distribution[1] on E, with parameters $\theta_s \in \Sigma_d$ to each word s of a c.s.d $\mathcal{D}$. Indeed, by denoting $\theta = (\theta_s)_{s\in\mathcal{D}}$ we define a conditional distribution on $\mathcal{X}$ which is the product of the likelihood of each transition contained in $\mathcal{X}$, namely:

$$P_{\mathcal{D},\theta}(X) = \prod_{i=1}^{N_X} \theta_{\mathcal{D}(x_c^i)}(x_l^i), \tag{3}$$

where for any word m in E^D, $\mathcal{D}(m)$ is the unique suffix of m in $\mathcal{D}$.

We present in Figure 1 an example where $E = \{A, B, C\}$, the maximal depth D is set to 3 and where $\mathcal{D} = \{A, AB, BB, ACB, BCB, CCB, C\}$, with corresponding θ_s parameters for $s \in \mathcal{D}$, each θ_s being a vector of the three-dimensional simplex Σ_3. We will also note $\mathcal{P}_D = \{(\mathcal{D}, \theta) : \mathcal{D} \in \mathcal{F}_D, \theta \in \Theta_\mathcal{D}\}$ the set of context-tree distributions of depth D.

C. Prior distributions on context-tree models

Having defined a family of distributions $\mathcal{P}_D$ and recalling (2), we define in this section a prior probability $\pi(\mathcal{D}, d\theta)$ on $\mathcal{P}_D$. This probability factorizes as $\pi(\mathcal{D}, d\theta) = \pi(\mathcal{D})\pi(d\theta|\mathcal{D})$, two terms which are defined as follows.

[1] Σ_d is the canonical simplex of dimension d, i.e. $\Sigma_d = \{\xi = (\xi_i)_{1\leq i\leq d} : \xi_i \geq 0, \sum \xi_i = 1\}$.

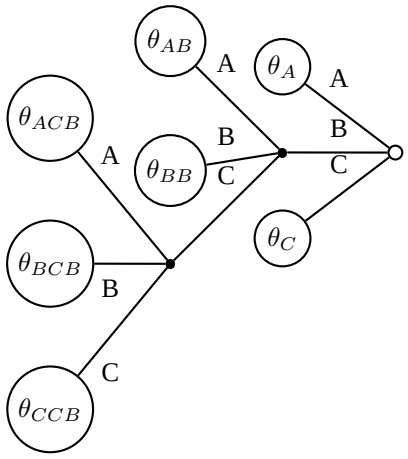

Fig. 1. Tree representation of a context-tree distribution

1) Prior on the tree structure: $\mathcal{F}_D$ is the set of complete trees of depth smaller than D. Intuitively it would make sense to put more prior weight on small trees than on large trees. Indeed, the number of different trees with a given number of leaves increases roughly exponentially with the number of leaves. As a result, small trees would have a very low influence compared to big trees if their prior probability was not boosted. Following [28] we define a simple probability π on $\mathcal{F}_D$ that has this property by describing a random generation of trees. Starting from the root, the tree generation process follows recursively the following rule: each node has d children with probability ϵ, and 0 children with probability $1-\epsilon$ (it is then a leaf). In mathematical terms, this defines a branching process. If we denote by $\mathring{\mathcal{D}}$ the strict suffixes of elements of $\mathcal{D}$, the probability of a tree is given by:

$$\pi(\mathcal{D}) = \prod_{s\in\mathring{\mathcal{D}}} \varepsilon \prod_{\substack{s\in\mathcal{D} \\ l(s)<D}} (1-\varepsilon) = \varepsilon^{\frac{|\mathcal{D}|-1}{d-1}} (1-\varepsilon)^{\mathrm{card}\{s\in\mathcal{D}|l(s)<D\}}. \tag{4}$$

2) Priors on multinomial parameters: For a given tree $\mathcal{D}$ we now define a prior on $\Theta_\mathcal{D} = (\Sigma_d)^\mathcal{D}$. We assume an independent prior among multinomials attached to different words with the following form:

$$\pi(d\theta|\mathcal{D}) = \prod_{s\in\mathcal{D}} \omega(d\theta_s).$$

Here ω is a prior distribution on the simplex Σ_d. Following [28] a simple choice is to take a Dirichlet prior of the form:

$$\omega_\beta(d\theta) = \frac{1}{\sqrt{d}} \frac{\Gamma(\sum_{i=1}^d \beta_i)}{\prod_{i=1}^d \Gamma(\beta_i)} \prod_{i=1}^d \theta_i^{\beta_i - 1} \lambda(d\theta),$$

where λ is Lebesgue's measure and $\beta = (\beta_i)_{i=1..d}$ is the parameter of the Dirichlet distribution. As it has been observed that mixtures of Dirichlet are a more natural way to model distributions on amino-acids [5], [18] we propose to use such a prior here. An additive mixture of n Dirichlet distributions is defined by n Dirichlet parameters $\beta^1, \ldots, \beta^n$ and by the probabilities $\gamma^1, \ldots, \gamma^n$ of each mixture (with $\sum_{k=1}^n \gamma^k = 1$),

and has the following definition:

$$\omega(d\theta_s) = \sum_{k=1}^{n} \gamma^k \omega_{\beta^k}(d\theta_s). \tag{5}$$

D. Triple mixture mutual information kernel

Combining the definition of the kernel (2) with the definition of the context-tree model distributions (3) and of the prior on the set of distributions (4, 5), we obtain the following expression for the kernel:

$$\mathcal{K}_\sigma(X,Y) = \sum_{\mathcal{D}\in\mathcal{F}_D} \pi(\mathcal{D}) \int_{\Theta_\mathcal{D}} P_{\mathcal{D},\theta}(X)^{\frac{\sigma}{N_X}} P_{\mathcal{D},\theta}(Y)^{\frac{\sigma}{N_Y}} \prod_{s\in\mathcal{D}} \omega(d\theta_s). \tag{6}$$

We observe that (6) involves three summations respectively over the trees (through prior π), the components of the Dirichlet mixtures (through weights γ), and the multinomial parameters (through ω_β priors). This generalizes the double mixture performed in [28] in the context of sequence compression by adding a mixture of Dirichlet, justified by our goal to process protein sequences.

IV. Kernel implementation

The definition of the kernel in (6) does not express a practical way to compute it. To do so, we propose to adapt the context-tree weighting algorithm, first introduced in [28], based on a factorization of the kernel along the branches of the context-tree. Let us introduce first a few more notations. We set, given $r \in \mathbb{N}$, $\beta = (\beta_i)_{1\le i\le r} \in (\mathbb{R}^{+*})^r$ and $\alpha = (\alpha_i)_{1\le i\le r} \in (\mathbb{R}^{+})^r$:

$$\mathbb{G}_\beta(\alpha) \stackrel{\text{def}}{=} \int_{\Sigma_r} \prod_{i=1}^{r} \theta_i^{\alpha_i} \omega_\beta(d\theta) = \frac{\Gamma(\beta_\cdot)}{\prod_{i=1}^r \Gamma(\beta_i)} \frac{\prod_{i=1}^r \Gamma(\alpha_i+\beta_i)}{\Gamma(\alpha_\cdot+\beta_\cdot)},$$

where Γ is the Gamma function, $\beta_\cdot = \sum_{i=1}^r \beta_i$, and $\alpha_\cdot = \sum_{i=1}^r \alpha_i$. The quantity $\mathbb{G}_\beta(\alpha)$ corresponds to the averaging of likelihoods $\mathbb{Q}_\theta(\alpha)$ under a Dirichlet prior of parameter β for θ varying in Σ_r. In the following implementation we assume that a numerical approximation for the function $\mathbb{G}_\beta$ is available. We can now divide the algorithm into two phases which can be computed alongside at each recursive step.

A. Defining counters

The first step of the algorithm is to compute, for $e \in E$ and $m \in E^D$, the following counters:

$$\rho_m(X) = \sum_{i=1}^{N_X} \mathbb{1}(x_c^i = m),$$

$$\hat\theta_{m,e}(X) = \begin{cases} \frac{\sum_{i=1}^{N_X} (x_c^i = m, x_l^i = e)}{\rho_m(X)} & \text{if } \rho_m(X) > 0 \\ \frac{1}{d} & \text{else} \end{cases},$$

$$a_{m,e}(X,Y) = \frac{\rho_m(X)}{|X|}\hat\theta_{m,e}(X) + \frac{\rho_m(Y)}{|Y|}\hat\theta_{m,e}(Y)$$

Counter $\rho_m(X)$ keeps track of the frequency of the counter m in the set X while $\hat\theta_{m,e}$ summarizes the empirical probability of the apparition of letter e after m has been observed. Finally $a_{m,e}(X,Y)$ takes into account a *weighted* average of the transitions encountered both in X and Y. To take into account smaller contexts we define the same values when m goes through M, the set of words of length less than D. The most efficient way to compute those counters is to start defining them when m only goes through visited contexts, which are up to N_X+N_Y, and then benefit from the following downward recursion on the length of the string m when m goes through all *suffixes* of visited contexts:

$$\rho_m(X) = \sum_{f\in E} \rho_{fm}(X),$$

$$\hat\theta_{m,e}(X) = \frac{\sum_{f\in E} \rho_{f.m}(X)\theta_{f.m,e}(X)}{\rho_m(X)},$$

$$a_{m,e}(X,Y) = \sum_{f\in E} a_{fm,e}(X,Y).$$

B. Recursive computation of the triple mixture

We can now attach to each m for which we have calculated the previous counters the value:

$$K_m(X,Y) = \sum_{k=1}^{n} \gamma^k \mathbb{G}_{\beta^k}\left(\sigma\left(a_{m,e}(X,Y)\right)_{e\in E}\right),$$

which computes two mixtures, the first being continuous on the possible values of θ weighted by a Dirichlet prior and the second being discrete by using the different weighted Dirichlet distributions given by the mixture (γ^k, β^k). By defining now the quantity $\Upsilon_m(X,Y)$, which is also attached to each visited word m and computed recursively:

$$\Upsilon_m(X,Y) = \begin{cases} K_m(X,Y) & \text{if } l(m) = D, \\ (1-\varepsilon)K_m(X,Y) & \\ \quad + \varepsilon \prod_{e\in E} \Upsilon_{e.m}(X,Y) & \text{if } l(m) < D \end{cases}.$$

We compute the third mixture over the different possible tree structures of our complete-suffix dictionary by taking into account the branching probability ε. Indeed, we finally have, recalling $\varnothing$ is the empty word, that:

$$\mathcal{K}_\sigma(X,Y) = \Upsilon_\varnothing(X,Y). \tag{7}$$

Proof: In order to prove (7), let us first fix a tree $\mathcal{D}$ and observe that, for $X = (x_c^i, x_l^i)_{i=1..N_X}$ and $Y = (y_c^i, y_l^i)_{i=1..N_Y}$:

$$\begin{aligned}
&\int_{\Theta_\mathcal{D}} P_{\mathcal{D},\theta}(X)^{\frac{\sigma}{N_X}} P_{\mathcal{D},\theta}(Y)^{\frac{\sigma}{N_Y}} \prod_{s\in\mathcal{D}} \left(\sum_{k=1}^{n} \gamma^k \omega_{\beta^k}(d\theta_s)\right) \\
&= \int_{\Theta_\mathcal{D}} \prod_{s\in\mathcal{D}} \left(\prod_{e\in E} \theta_s(e)^{\sigma a_{s,e}(X,Y)} \left(\sum_{k=1}^{n} \gamma^k \omega_{\beta^k}(d\theta_s)\right)\right) \\
&= \prod_{s\in\mathcal{D}} \sum_{k=1}^{n} \gamma^k \int_{\Sigma_d} \left(\prod_{e\in E} \theta_s(e)^{\sigma a_{s,e}(X,Y)} \omega_{\beta^k}(d\theta_s)\right) \\
&= \prod_{s\in\mathcal{D}} \sum_{k=1}^{n} \gamma^k \mathbb{G}_{\beta^k}\left(\sigma\left(a_{s,e}\left(X,Y\right)\right)_{e\in E}\right) = \prod_{s\in\mathcal{D}} K_s(X,Y),
\end{aligned}$$

where we have used Fubini's theorem to factorize the integral in the second line. Having in mind (6), we have thus proved that $\mathcal{K}_\sigma(X,Y) = \sum_{\mathcal{D}\in\mathcal{F}_D} \pi(\mathcal{D}) \prod_{s\in\mathcal{D}} K_s(X,Y)$. The second

part of the proof is identical to the one given in [28] [6] to which we refer to finalize this result. ■

The computation of the counters has a linear cost in time and memory with respect to $N_X + N_Y$. As only nodes that correspond to suffixes of X and Y are created, recursive computation of Υ_m is also linear (the values Υ_m on non-existing nodes being equal to 1). As a result, the computation of the kernel is linear in time and space with respect to $N_X + N_Y$.

V. Redundancy analysis

As explained previously, our kernel actually considers a sequence as a set of weighted empirical distributions $\{(\rho_m, \hat{\theta}_m)\}_{m \in M}$. These couples are actually used to compute the likelihood of such a set with respect to a specific context-tree distribution $(\mathcal{D}, \theta)$ contained in the manifold of all distributions defined by model $\mathcal{D}$. This manifold is a submanifold of $(\Sigma_d)^M$ which admits the family of multinomial parameters $(\theta_s)_{s \in M}$ as a coordinate system. The elements $\{\hat{\theta}_s, s \in \mathcal{D}\}$ can thus be seen as the coordinates of X in the submanifold associated with model $\mathcal{D}$ and weights ρ_s can be seen as the empirical measure of each $\hat{\theta}_s$ present in X.

We denote by $\mathrm{kl}(\theta||\theta')$ the kullback-leibler divergence between θ and θ', two multinomial parameters of size d, i.e $\mathrm{kl}(\theta||\theta') = \sum_{i=1..d} \theta_i \ln \frac{\theta_i}{\theta'_i}$. We also note $\mathcal{H}(\theta)$ the entropy of θ, i.e. $\mathcal{H}(\theta) = \sum_{i=1..d} \theta_i \ln \theta_i$. The mixture coding probability P_π on $\mathcal{X}$ following the π prior on $\mathcal{F}_D$ can be rewritten as a simple function of ρ and $\hat{\theta}$:

$$P_\pi(\rho, \hat{\theta}) = \sum_{\mathcal{D} \in \mathcal{F}_D} \pi(\mathcal{D}) \prod_{s \in \mathcal{D}} e^{-\sigma \rho_s \mathcal{H}(\hat{\theta}_s)} \int_{\Sigma_d} e^{-\sigma \rho_s kl(\hat{\theta}_s||\theta)} \omega(d\theta)$$

We consider $r_\pi \stackrel{\text{def}}{=} -\ln P_\pi$, the redundancy of the coding probability computed by this mixture. This quantity can be interpreted to express the value of our kernel by defining the function t_π which measures the convexity of r_π on $\Sigma_{|\mathcal{D}|} \times \Theta_{\mathcal{D}}$:

$$\begin{aligned} t_\pi(X, Y) = & \frac{1}{2} \left[r_\pi \left(\hat{\rho}(X), \hat{\theta}(X) \right) + r_\pi \left(\hat{\rho}(Y), \hat{\theta}(Y) \right) \right] \\ & - r_\pi \left(\frac{\hat{\rho}(X) + \hat{\rho}(Y)}{2}, \frac{\hat{\theta}(X) + \hat{\theta}(Y)}{2} \right), \end{aligned}$$

where we have used the notation $\hat{\rho}(X) = \frac{1}{|X|} \rho(X)$. Finally we have, by defining the renormalized kernel $\tilde{K}_\sigma$ as

$$\tilde{\mathcal{K}}_\sigma(X, Y) = \mathcal{K}_\sigma(X, Y) / \sqrt{\mathcal{K}_\sigma(X, X) \mathcal{K}_\sigma(Y, Y)},$$

that

$$\tilde{\mathcal{K}}_\sigma(X, Y) = e^{-t_\pi(X, Y)},$$

providing us with a geometrical interpretation, in terms of convexity of the redundancy function, of the value computed by our kernel.

VI. Experiments

We report preliminary results concerning the performance of the MI kernel on a widely used benchmark experiment proposed in [13] which tests the capacity of SVMs to detect remote homologies between protein domains. This is simulated by recognizing domains that are in the same SCOP[12] (ver. 1.53) superfamily, but not in the same family, using the procedure described in [13]. We used the files compiled by the authors of [19]. For each of the 54 families tested, we computed the ROC (Receiving Operator Characteristic) to measure the performance of a SVM based on the MI kernel (the ROC score is the normalized area under the curve which plots the number of true positives as a function of false positives). We tested different parameters of our kernel, and compared its performance with the mismatch kernel presented in [16], which performed state-of-the-art accuracy level when published and can also be implemented in *linear time*. The classification and results were led using the publicly available Gist 2.0.5 implementation of SVM[2], applying a 2-norm soft margin by adding a diagonal factor to the kernel matrix equal to the exact proportion of positives in the dataset (diagonal factor of one) without any specific tuning of parameters.

Our kernel has several parameters. The depth D, the width σ and the branching probability ε are the most elementary to play with; the selection of a Dirichlet mixture is a more difficult choice. Given the large number of parameters and the risk of overfitting the benchmark dataset by carefully optimizing them, we only report preliminary results with two settings. First we used a single Dirichlet distribution with parameters $1/2, \ldots, 1/2$ (known as the Jeffrey or the Krichevski-Trofimov prior [28]), with $D = 5$, $\sigma = 5$, $\varepsilon = 0.5$. Second, we used a basic 3 component Dirichlet mixture that models three classes of amino-acids (hydrophobic/hydrophilic/highly conserved). This mixture, called `hydro-cons.3comp`, was downloaded from a Dirichlet mixture repository[3]. Other parameters were set to $D = 4$, $\sigma = 1$ and $\varepsilon = 0.5$.

Figure 2 plots the total number of families for which a given methods exceeds a ROC score threshold. There is no significant difference between the three methods. The mismatch kernel seems to perform better on families with large ROC, while the MI kernels tend to outperform the mismatch kernel for families with a ROC below 0.85. This observation is encouraging as it suggests that MI kernels might be better adapted to difficult problems, corresponding to low sequence similarity, than the mismatch kernel, although our kernel is only based on the same features as the spectrum kernel [15] which is known to perform worse than the mismatch kernel tested.

VII. Conclusion

We introduced a novel class of kernels for sequences that are fast to compute and have the flexibility to include prior knowledge through the definition of probabilistic models and

[2] `http://microarray.cpmc.columbia.edu/gist/download.html`
[3] `http://www.cse.ucsc.edu/research/compbio/dirichlets/`

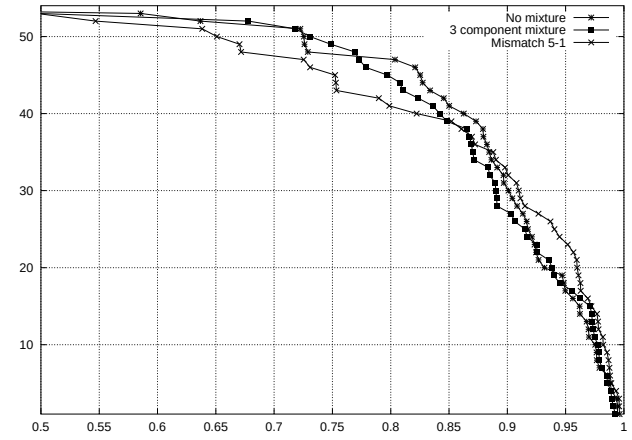

Fig. 2. Performance of three kernels on the problem of recognizing domain's superfamily. The curve shows the total number of families for which a given methods exceeds a ROC score threshold.

prior distribution. The kernel is a mutual infofmation kernel based on a family of context-tree models, and makes a link between the string kernels and the theory of universal source coding. On a benchmark experiment of remote homology detection it performs at a state-of-the-art level. Further accuracy improvements are expected from a more careful tuning of the parameters, on the one hand, and from the implementation of sampling strategies to derive extended sets of transitions X from a single sequence m_x by incorporating mismatches for instance.

VIII. Acknowledgments

The authors would like to thank anonymous reviewers for their remarks as well as Tatsuya Akutsu, Hiroto Saigo, Hiroyuki Nakahara and Jérémie Jakubowicz for fruitful discussions.

References

[1] G. Bejerano and G. Yona. Modeling protein families using probabilistic suffix trees. In S. Istrail, P. Pevzner, and M. Waterman, editors, *Proceedings of the3rd Annual International Conference on Computational Molecular Biology (RECOMB)*, pages 15–24, Lyon, France, 1999. ACM Press.

[2] A. Ben-hur and D. Brutlag. Remote homology detection: a motif based approach. *Bioinformatics*, 2003. To appear.

[3] C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Springer-Verlag, 1984.

[4] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th annual ACM workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

[5] M. P. Brown, R. Hughey, A. Krogh, I. S. Mian, K. Sjölander, and D. Haussler. Using Dirichlet mixture priors to derive hidden Markov models for protein families. In *Proc. of First Int. Conf. on Intelligent Systems for Molecular Biology*, pages 47–55, Menlo Park, CA, 1993. AAAI/MIT Press.

[6] O. Catoni. *Statistical learning theory and stochastic optimization, Saint-Flour lecture notes*. Springer Verlag, *to appear*.

[7] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

[8] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis - Probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge, UK, 1998.

[9] E. Eskin, W. Noble, and Y. Singer. Protein family classification using sparse markov transducers. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, August 2000.

[10] D. Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz, 1999.

[11] S. Hua and Z. Sun. Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 17(8):721–728, 2001.

[12] T. Hubbard, A. Murzin, S. Brenner, and C. Chothia. Scop: a structural classification of proteins database, 1997.

[13] T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 7(1,2):95–114, 2000.

[14] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In T. Faucett and N. Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*, pages 321–328. AAAI Press, 2003.

[15] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: a string kernel for svm protein classific ation. In R. B. Altman, A. K. Dunker, L. Hunter, K. in Lauerdale, and T. E. Klein, editors, *Proceedings of the Pacific Symposium on Biocomputing 2002*, pages 564–575. World Scientific, 2002.

[16] C. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for svm protein classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[17] L. Liao and W. S. Noble. Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In *Proceedings of the Sixth Annual International Conference on Computational Molecular Biology*, pages 225–232, 2002.

[18] I. Nemenman, F. Shafee, and W. Bialek. Entropy and inference, revisited. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[19] W. S. Noble and L. Liao. Combining pairwise sequence similarity and support vector machines for remote protein homology detection. *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology*, pages 225–232, 2002.

[20] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization , Optimization, and Beyond*. MIT Press, Cambridge, MA, 2002.

[21] B. Schölkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA, 2004. To appear.

[22] B. Schölkopf, J. Weston, E. Eskin, C. Leslie, and W. S. Noble. A kernel approach for learning from almost orthogonal patterns. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proceedings of ECML 2002, 13th European Conference on Machine Learning, Helsinki, Finland, August 19-23, 2002*, volume 2430 of *Lecture Notes in Computer Science*, pages 511–528. Springer, 2002.

[23] M. Seeger. Covariance kernels from bayesian generative models. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA. MIT Press.

[24] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New-York, 1998.

[25] J.-P. Vert. A tree kernel to analyze phylogenetic profiles. *Bioinformatics*, 18:S276–S284, 2002.

[26] J.-P. Vert, H. Saigo, and T. Akutsu. Local alignment kernels for protein sequences. In B. Scholkopf, K. Tsuda, and J.-P. Vert, editors, *Kernel Methods in Computational Biology*. MIT Press, 2004.

[27] C. Watkins. Dynamic alignment kernels. In A. Smola, P. Bartlett, B. Schölkopf, and D. S. rmans, editors, *Advances in Large Margin Classifiers*, pages 39–50. MIT Press, Cambridge, MA, 2000.

[28] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens. The context-tree weighting method: basic properties. *IEEE Transancations on Information Theory*, pages 653–664, 1995.