Sequence Discrimination Using Phase-Type Distributions

Jérôme Callut[1,2] and Pierre Dupont[1,2]

[1] Department of Computing Science and Engineering, INGI
Université catholique de Louvain,
Place Sainte-Barbe 2,
B-1348 Louvain-la-Neuve, Belgium
{Jerome.Callut, Pierre.Dupont}@uclouvain.be
[2] UCL Machine Learning Group
http://www.ucl.ac.be/mlg/

Abstract. We propose in this paper a novel approach to the classification of discrete sequences. This approach builds a model fitting some dynamical features deduced from the learning sample. These features are discrete *phase-type* (PH) distributions. They model the first passage times (FPT) between occurrences of pairs of substrings. The PHit algorithm, an adapted version of the Expectation-Maximization algorithm, is proposed to estimate PH distributions. The most informative pairs of substrings are selected according to the Jensen-Shannon divergence between their class conditional empirical FPT distributions. The selected features are then used in two classification schemes: a maximum a posteriori (MAP) classifier and support vector machines (SVM) with marginalized kernels. Experiments on DNA splicing region detection and on protein sublocalization illustrate that the proposed techniques offer competitive results with smoothed Markov chains or SVM with a spectrum string kernel.

Keywords: Supervised sequence classification, Markov chains, First passage times, Expectation-Maximization, Jensen-Shannon divergence.

1 Introduction

This paper is concerned with a supervised classification problem in which the instances are sequences defined over a discrete alphabet. Practical applications of this task range from the recognition of boundaries between introns and exons in DNA sequences to musical pieces classification.

The approach proposed in this paper relies on building a model to fit some dynamical features in the sample. In this context, a former technique was based on the *mean first passage times* between individual symbols [3]. In this paper, we focus not only on the mean but on the complete distribution of the times between occurrences of substrings in the sample. More precisely, given a pair of substrings (v, w) called here a *feature* of the sequence to be classified, we are looking at the number of steps taken to observe the next occurrence of w after having observed v. The distribution of these measures forms the First Passage Time (FPT) dynamics of a sequential process with respect to the feature (v, w). The purpose of this paper is to exploit the different FPT dynamics between the classes to perform sequence classification. Since the number of features can be potentially large, only a restricted number of the observed features are considered. The selection of the features (v, w) is performed using the Jensen-Shannon (JS) divergence [8]. Given an observed feature (v, w), the empirical FPT distribution is estimated for each class. The JS divergence is then applied to rank the considered features. The features offering the largest JS divergence between their class conditional distributions are kept for the classification process. Once the features have been selected, the associated FPT dynamics are modeled with discrete phase-type distributions.

Discrete phase-type distributions (PH) form a broad class of distributions that generalize the family of negative binomial distributions and have applications in various stochastic models such as queuing systems [7]. A PH distribution can be defined as the distribution of the time to absorption in an absorbing Markov chain (MC). Our first contribution is an EM algorithm for fitting discrete PH distributions. Our algorithm can be considered as an adaptation to discrete distributions of the work of Asmussen and Olsson [1], which handles continuous PH distributions. Modeling the FPT dynamics with PH distributions allows one to control the generalization, that is, the probability mass given to unseen events, by tuning the number of phases (see section 2). In this sense, the PH modeling can be thought of as a smoothing technique of the empirical FPT distributions observed in the sequences. The estimated PH distributions are used to solve the sequence classification problem. Two classification schemes are considered: a maximum a posteriori (MAP) classifier and support vector machines (SVM).

The rest of this paper is organized as follows. Section 2 reviews standard Markov chains and PH distributions. Section 3 introduces the PHit algorithm for fitting discrete PH distributions. Section 4 describes the feature selection procedure and presents a MAP classifier as well as a kernel based on PH distributions. Finally, section 5 shows experimental results obtained with the proposed techniques applied to DNA splicing junction detection and protein sublocalization.

2 Discrete PH Distributions

The methods proposed to solve the sequence classification problem in section 4 rely on the first passage times (FPT) between events in sequences (see definition 2). These times can be conveniently modeled by PH distributions introduced hereafter. For a detailed introduction to the MC theory and to PH distributions, the reader is respectively referred to the classical text books [6] and [7]. A MC can be represented by a 3-tuple $M = \langle Q, A, \boldsymbol{\iota} \rangle$ where Q is a finite set of states of size m, A is an $m \times m$ stochastic transition matrix and $\boldsymbol{\iota}$ is a $1 \times m$ vector representing the initial probability distribution. In a MC, a state q is said to be *absorbing* if there is a probability 1 to go from q to itself. In other words, once an absorbing state has been reached, the process will stay on this state forever. A MC for which there is a probability 1 to end up in an absorbing state is called an *absorbing MC*. In such a model, the state set can be divided into the absorbing state set Q_A and its complementary set, the transient state set Q_T. An absorbing MC with a single absorbing state will be called a *reduced* absorbing MC (or a *reduced* MC for short). A fundamental characteristic of absorbing MC is the *time to absorption*, *i.e.* the number of steps the process takes to reach an absorbing state. The distribution of the time to absorption is of *phase-type*.

Definition 1 (Discrete Phase-type Distribution). *A probability distribution $\varphi(.)$ on $\mathbb{N}^0$ is a* distribution of phase-type *if and only if it is the distribution of the time to absorption in an absorbing MC.*

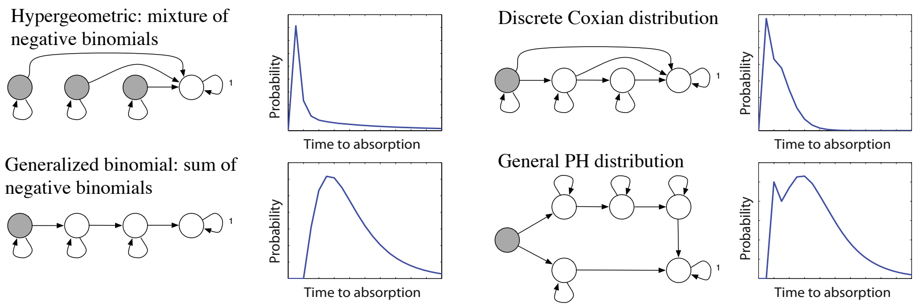
It should be pointed out that it is always possible to transform an absorbing MC with several absorbing states into a reduced one with the same distribution of the time to absorption. Hence, without loss of generality, we will only consider reduced MC in normal form, the last state being absorbing :

$$\boldsymbol{\iota} = (\boldsymbol{u} \quad 0), \quad A = \begin{pmatrix} T & \boldsymbol{e} \\ \boldsymbol{0} & 1 \end{pmatrix}$$

where $\boldsymbol{u}$ is a $1 \times (m-1)$ *initial vector* for transient states, T is an $(m-1) \times (m-1)$ matrix called the *phase generator*, $\boldsymbol{e}$ is an $(m-1) \times 1$ vector called the *absorption vector* and $\boldsymbol{0}$ is an $1 \times (m-1)$ vector of zeros. It will be assumed that the process always starts in a non-absorbing state: $\iota_m = 0$. A PH distribution φ is completely determined[1] by a pair $(\boldsymbol{u}, T)$ which is called the *representation* of the distribution. The probability distribution of $\varphi(.)$ is given by $\varphi(k) = \boldsymbol{u}T^{k-1}\boldsymbol{e}$ for all $k \geq 1$ which means that the probability of being absorbed in k steps is the probability of starting in any transient state, then to move over transient states during $k-1$ steps, and finally to get absorbed. Each transient state of the representing MC is called a *phase*. This technique is powerful since it allows one to decompose complex distributions as a combination of phases. For instance, the class of PH distributions contains the negative binomial, the hyper-geometric and the discrete Coxian distributions, to name a few. These distributions can be instantiated using particular absorbing MC structures. This point is illustrated in figure 1. A distribution with an initial vector and a transition matrix with no particular structure is called here a *general PH distribution*.

The next section presents a tool for estimating a PH distribution from a data sample. In this context, tuning the number of phases allows one to deal with the *bias-variance* trade-off. Indeed, fitting a distribution using a few phases gives an important probability mass to unseen events, which are unseen first passage times between two substrings in our context, while the overfitting is likely to happen when using a large number of phases.

[1] Since the matrix A is stochastic, the vector $\boldsymbol{e}$ can be obtained from the matrix T.



Hypergeometric: mixture of negative binomials
Discrete Coxian distribution
Generalized binomial: sum of negative binomials
General PH distribution
Probability
Time to absorption
1

Fig. 1. Different kinds of PH distributions and associate absorbing MC structures. The process has a strictly positive probability to start in states filled with gray and a null probability to start in the other states.

3 Fitting Discrete PH Distributions: PHit

In this section, we introduce the PHit algorithm for fitting discrete PH distributions. This algorithm is used to compute the features in the classification methods presented in section 4. Here, the samples do not directly correspond to the learning sequences of the classification task but to discrete times between two particular substrings in these sequences.

The EMpht algorithm, developed by Asmussen and Olsson [1], fits continuous PH distributions. In contrast, we focus here on discrete PH distributions. In particular PHit deals with negative binomial state duration distribution in an absorbing MC while these durations are modeled in EMpht by a negative exponential density, typical of continuous Markov processes. The re-estimation formula in both algorithms are thus distinct. Bobbio *et al.* [2] also proposed a technique for fitting discrete PH distributions, however it is restricted to a particular class of PH distributions (acyclic PH distributions) while PHit can deal with general PH distributions.

Given a set of l observations (times between two events) $Z = \{z_1, \dots, z_l\}$ with $z_i \in \mathbb{N}^0$ and a number m of states, PHit estimates the parameters $(\boldsymbol{u}, T)$ of a PH distribution φ with $m-1$ phases that maximize the likelihood with respect to Z. To do so, the iterative Expectation-Maximization (EM) algorithm is used [4]. The basic idea is to consider each z_i as an incomplete observation of the underlying process $\{X_t\}$ (an absorbing MC). More precisely, we observe the time to absorption of a process realization but not the sequence of states reached by the process during this realization. Let $H = \{h_1, \dots, h_l\}$ be the set of hidden sequences relative to Z, where h_i is a sequence of z_i (transient) states. The likelihood of φ with respect to one complete observation (z, h) is given by $P[(z,h) \mid \varphi] = u_{h_1} \left(\prod_{i=1}^{z-1} T_{h_i,h_{i+1}}\right) e_{h_z}$. We introduce below three kinds of auxiliary variables which are useful in the estimation process:

- $S(i)$: the number of observations in H starting in state i.
- $F(i)$: the number of observations in H ending in state i.

– $N(i,j)$: the number of times state j immediately follows state i in H.

The likelihood of φ with respect to the set of all observations is defined as:

$$P((Z,H) \mid \varphi) = \prod_{i=1}^{m-1} u_i^{S(i)} e_i^{F(i)} \prod_{j=1}^{m-1} T_{ij}^{N(i,j)}$$

The EM algorithm finds the maximum likelihood estimates of the parameters λ of a joint distribution $P(Z,H|\lambda)$ when the variable H is unobserved (H is a *latent* or *hidden* variable). This is achieved by computing iteratively the parameters λ that maximize $E[\ln(P(Z,H \mid \lambda)) \mid Z]$. Each EM iteration involves two steps: (i) the computation of the conditional expectation of the latent variables given the last parameter values λ^{t-1} and the partial observations Z, (ii) the maximization of the parameters λ^t given the conditional expectation of the latent variables. The computations of these two steps in the PHit algorithm are detailed hereafter.

Expectation step

In PHit, the latent variables are the auxiliary variables $S(i)$, $F(i)$ and $N(i,j)$. Their conditional expectations are conveniently computed using *forward* variables defined as $\alpha_{ij}(t) = P[X_t = j \mid X_0 = i]$ for all $1 \leq i < m, 1 \leq j \leq m$ and $t \geq 0$. That is, $\alpha_{ij}(t)$ is the probability of being in state j after t steps while starting from state i. The forward variables can be computed using the following recurrence:

$$\alpha_{ij}(0) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \alpha_{ij}(t) = \sum_{k=1}^{m-1} \alpha_{ik}(t-1).A_{k,j}$$

In the sequel, the shorthand $\alpha_j(t)$ stands for $\sum_{i=1}^{m-1} u_i \alpha_{ij}(t)$, which is the probability of being in state j after t steps when starting according to the initial distribution, $\alpha_m(t) = \varphi(t)$ being a special case. The forward variables $\alpha_j(t)$ can be thought of as a simplification of the forward variables used in the Baum-Welch algorithm which estimates the parameters of Hidden Markov Models (HMMs) [9]. Indeed, in the Baum-Welch algorithm, a forward variable computes the probability of being in a state after having accepted a given string. In PHit, a forward variable computes the probability of being in a given state after t steps no matter how this state is reached.

Conditional expectation of $S(i)$: The variables $S(i)$ can be decomposed as a sum of indicator variables[2], $S(i) = \sum_{k=1}^{l} \mathbb{1}\{h_{k,1} = i\}$ where the notation $h_{k,j}$ stands for the j-th state in the k-th observation. The conditional expectation $E[S(i) \mid z_1, \dots, z_l]$ is given by

$$\begin{aligned} E[S(i) \mid z_1, \dots, z_l] &= E[\sum_{k=1}^{l} \mathbb{1}\{h_{k,1} = i\} \mid z_1, \dots, z_l] = \sum_{k=1}^{l} E[\mathbb{1}\{h_{k,1} = i\} \mid z_k] \\ &= \sum_{k=1}^{l} P[h_{k,1} = i \mid z_k] \end{aligned}$$

[2] The value of an indicator variable $\mathbb{1}\{X = x\}$ is 1 when the random variable X equals x and 0 otherwise. The expectation of an indicator variable is simply $E[\mathbb{1}\{X = x\}] = P[X = x]$.

The conditional probability $P[h_{k,1} = i \mid z_k] = \frac{P[h_{k,1}=i, z_k]}{P[z_k]}$ with $P[h_{k,1} = i, z_k] = u_i \alpha_{im}(z_k)$ and $P[z_k] = \alpha_m(z_k)$. Finally, the conditional expectation is defined as

$$E[S(i) \mid z_1, \ldots, z_l] = \sum_{k=1}^{l} \frac{u_i \alpha_{im}(z_k)}{\alpha_m(z_k)}$$

Conditional expectation of $N(i,j)$: The variables $N(i,j)$ can be decomposed as $\sum_{k=1}^{l} \sum_{d=1}^{z_k-1} \mathbb{1}\{h_{k,d} = i \wedge h_{k,d+1} = j\}$, that is, counting the presence of the transition $i \to j$ in each position of each hidden sequence. By the same reasoning as above, one obtains

$$E[N(i,j) \mid z_1, \ldots, z_l] = \sum_{k=1}^{l} \sum_{d=1}^{z_k-1} P[h_{k,d} = i \wedge h_{k,d+1} = j \mid z_k]$$

The joint probability $P[h_{k,d} = i \wedge h_{k,d+1} = j, z_k] = \alpha_i(d-1) T_{ij} \alpha_{jm}(z_k - d)$, that is the probability of starting in any transient state, to reach state i after $d-1$ steps, then to perform the transition $i \to j$, and finally to get absorbed $z_k - d$ steps later. It follows that

$$E[N(i,j) \mid z_1, \ldots, z_l] = \sum_{k=1}^{l} \sum_{d=1}^{z_k-1} \frac{\alpha_i(d-1) T_{ij} \alpha_{jm}(z_k - d)}{\alpha_m(z_k)}$$

Conditional expectation of $F(i)$: The variables $F(i)$ can be decomposed as $F(i) = \sum_{k=1}^{l} \mathbb{1}\{h_{k,z_k} = i\}$. By the same reasoning as above, the conditional expectation is $E[F(i) \mid z_1, \ldots, z_l] = \sum_{k=1}^{l} P[h_{k,z_k} = i \mid z_k]$. The joint probability $P[h_{k,z_k} = i, z_k] = \alpha_i(z_k - 1) e_i$, that is the probability of starting in any transient state, to reach state i after $z_k - 1$ steps then to get absorbed. The conditional expectation is given by

$$E[F(i) \mid z_1, \ldots, z_l] = \sum_{k=1}^{l} \frac{\alpha_i(z_k - 1) e_i}{\alpha_m(z_k)}$$

Maximization step

Once the expected values of the latent variables have been computed, the maximum likelihood estimators of the PH distribution parameters are

$$u_i = \frac{\overline{S(i)}}{\sum_{k=1}^{m-1} \overline{S(k)}}, e_i = \frac{\overline{F(i)}}{\sum_{k=1}^{m-1} \overline{F(k)}} \text{ and } T_{ij} = \frac{\overline{N(i,j)}}{\sum_{k=1}^{m-1} \overline{N(i,k)}} \quad \forall i, j, 1 \leq i, j < m$$

where the bar notation stands for the conditional expectation of the variables.

The parameters $(\boldsymbol{u}, T)$ are initialized at random at the beginning of the algorithm. PHit iterates until the relative likelihood improvement falls below a user-defined threshold. In our experiments, PHit is rather insensitive to the initialization. In other words, the value of the likelihood obtained after convergence is rarely affected by different initializations. However, the parameters found by PHit can be different using different initializations since the representation of a PH distribution is not unique. As in the Baum-Welch algorithm, the forward variables can efficiently be computed by use of a lattice data structure. More precisely, for each transient state i a lattice is built to compute the forward variables $\alpha_{ij}(t)$ with $1 \leq j \leq m$. The time complexity of building such a lattice is $\mathcal{O}(L.m^2)$ where L is the longest absorption time in Z. The time complexity is $\mathcal{O}(L.m^3 + L.l.m^2)$ for the expectation step and $\mathcal{O}(m^2)$ for the maximization step. PHit has been implemented in the ANSI C language and it can estimate PH distributions with 100 phases over 2,000 events in about one minute on a standard PC.

4 Classification Using PH Distributions

In section 2, we argue that the time dynamics between events in a sequential process are conveniently modeled by PH distributions. In practice, these distributions can be estimated from a data sample by use of the PHit algorithm presented in section 3. In the present section we describe how PH distributions are used to solve a supervised classification problem in which the instances are sequences defined over a discrete alphabet Σ and the labels belong to a set $\mathcal{Y}$. Given a set of n examples $\{(s_1, y_1), \ldots, (s_n, y_n)\}$ where $s_i \in \Sigma^*$ is a sequence and $y_i \in \mathcal{Y}$ its label (or its class), one wants to estimate a function $f : \Sigma^* \rightarrow \mathcal{Y}$ for predicting the label of new sequences. Our methods run into 3 steps: (i) features selection, (ii) choosing the number of phases and modeling the selected features with PH distributions, (iii) classifier training. The features extracted from the learning sequences are the first passage times between pairs of events (occurrences of substrings).

Definition 2 (First Passage Times between a pair of events in a sequence). *Given a sequence s defined on an alphabet Σ and two substrings $v, w \in \Sigma^+$. For each occurrence of v in s, the first passage time to w is defined as the finite number of steps taken before observing the next occurrence of w. The* first passage times *from v to w in s is a multiset defined as the first passage times to w for all occurrences of v in s.*

For instance, let us consider the alphabet $\Sigma = \{a, b\}$, the sequence $s = aababba$ and the events $v = ab$ and $w = ba$. The value of the feature (ab, ba) is $\phi_{(ab,ba)}(s) = \{3, 1\}$. Let us note that the step count starts after the last character of v and it does not take the length of w into account. The choice of the features (pairs of events) is an important step in the classification process. The potential number of features is bounded by $(\sum_{i=1}^{N} |\Sigma|^i)^2 \in \mathcal{O}(|\Sigma|^{2N})$, where $|\Sigma|$ denotes the alphabet size. However the number of features observed in practice is often below this bound. It can be shown that an alternative upper bound is $n.L^2.K^2$, where L is the length of the longest sequence in the training set containing n examples and K is the maximal length of the considered substrings (in our experiments, $K = 3$).

Since our classifications methods rely on the difference between the class conditional FPT dynamics, the proposed features selection procedure extracts the features for which the empirical FPT distribution differs the most among the classes. To do so, the generalized Jensen-Shannon (JS) divergence is used [8]. A larger JS divergence between the empirical distributions of various samples indicates that they are more likely to have been drawn from different source distributions (i.e. from different classes). This measure is defined as follows.

$$JS(P_1, \ldots, P_r) = H\left(\sum_{i=1}^r \pi_i H(P_i)\right) - \sum_{i=1}^r \pi_i H(P_i)$$

where $P_1, \ldots, P_r$ are distributions, $\pi_1, \ldots, \pi_r$ are strictly positive weights summing up to one and $H(P) = -\sum_{x \in \Omega} P[x] \log P[x]$ is the Shannon entropy. The JS divergence has several advantages over the Kullback-Leibler (KL) divergence [8], a classical measure to compare two distributions: (i) it can be applied to more than two distributions, (ii) the relative importance of the distributions can be parametrized with weights (in our experiments, uniform weights are used), (iii) it can be thought as a symmetrized and smoothed (it is relative to the mean of the distributions) variant of the KL divergence, (iv) when applied to two distributions, the square root of the JS divergence enjoys the properties of a true distance metric. For each class, the empirical FPT distributions between every observed event pairs are computed. The score of a feature is defined as the JS divergence between the empirical FPT distributions of each class, weighted by the prior probability of the feature. The prior probability of a feature (v, w) is defined as $P[(v, w)] = \frac{\mathrm{C}(v,w)}{\sum_{v',w' \in \Sigma^{\leq K}} \mathrm{C}(v',w')}$, where $\mathrm{C}(v, w)$ is the number of times a string v is followed by a string w in the training set and $\Sigma^{\leq K}$ is the set of non-empty strings up to length K defined on the alphabet Σ. The features are ranked with respect to their score and the highest ranked features are kept for the classification process. In the sequel, the set of selected features will be denoted by $\mathcal{F}$. In practice, a set of 10^5 features can be ranked in about 30 seconds on a standard PC.

4.1 Maximum a Posteriori (MAP) Classifier

In this section, we introduce a maximum a posteriori (MAP) classifier based on PH distributions. Once the features have been selected, the related FPT dynamics are modeled with PH distributions for each class. The notation $\varphi^y_{(v,w)}(.)$ stands for the PH distribution relative to (v, w) estimated from the sequences of the class y. Our classifier makes the assumption that the features are independent. As usual for models making this naive assumption, the independence is not always satisfied but good results are obtained in practice. Consequently, the likelihood of a class y with respect to a sequence s is computed as $P[s\,|\,y] = \prod_{(v,w) \in \mathcal{F}} P[\phi_{(v,w)}(s)\,|\,y]$, where $P[\phi_{(v,w)}(s)\,|\,y] = \prod_{z \in \phi_{(v,w)}(s)} \varphi^y_{(v,w)}(z)$. Predicting the label of a sequence s is made by selecting the class that maximizes the posterior probability $\hat{y} = \mathrm{argmax}_y\, P[s\,|\,y]P[y]$, where $P[y]$ denotes the prior probability of the class y.

4.2 SVM in the PH Feature Space

We introduce here the PH kernel which maps the sequences in a feature space based on PH distributions. For each feature (v, w), a *marginalization kernel* [10] $k_{(v,w)}(.,.)$ computing the probability that two sequences have been generated together is introduced as follows.

$$k_{(v,w)}(s, s') = P_{(v,w)}[s, s'] = \sum_{y \in \mathcal{Y}} P[\phi_{(v,w)}(s) \mid y] P[\phi_{(v,w)}(s') \mid y] P[y]$$

The PH kernel, relative to the complete feature set $\mathcal{F}$, is defined as

$$k(s,s') = \sum_{(v,w)\in\mathcal{F}} P_{(v,w)}[s,s'] = \sum_{(v,w)\in\mathcal{F}} k_{(v,w)}(s,s')$$

The PH kernel amounts to compute a dot product in the space where a sequence s is mapped to $\left(\sqrt{P[y]}.P[\phi_{(v,w)}(s) \mid y]\right)_{y\in\mathcal{Y},(v,w)\in\mathcal{F}}$. The PH distributions used to compute the probabilities $P[\phi_{(v,w)}(s)|y]$ are estimated from a part of the training data (80 % in our experiments) and the rest of the data is used to train the SVM. The rationale is that the training data are no longer independent if they are used to build the kernel mapping. Once the SVM has been trained, new sequences are classified by looking at which side of the hyperplane they lie in the PH feature space.

5 Experiments

This section presents the experimental results obtained for two classification tasks: (i) DNA splicing region detection (`Splice` dataset) and (ii) protein sublocalization (`DBSubloc` database). The `Splice` dataset[3] is made of windows of 60 symbols from DNA sequences containing intro-exon (IE) or exon-intron (EI) boundaries or neither of them. We restrict here our attention to binary classification by considering sequences labeled either EI or IE. The class priors are equal and the training, validation and test sets contain respectively 975, 253 and 253 sequences. The `DBSubloc` database[4] contains protein sequences (primary structures) with their subcellular localization for various organisms. The classification tasks considered here consists in finding if a protein from a plant organism is located in the membrane or in the mitochondria of the cell. The average length of the sequences is 406. The class priors are respectively 0.45 and 0.55 for the membrane and the mitochondria classes and the training, validation and test sets contain respectively 151, 51 and 50 sequences.

The influence of the parameters (the number of phases and the number of features) is evaluated with both classification methods using the `Splice` validation data. Figure 2 presents learning curves using training data of growing sizes. For each size (except for 100%), 10 samples have been randomly extracted from the training set in order to produce averaged results with standard deviations. The left side of Figure 2 shows the accuracy obtained on validation data using the MAP classifier with increasing number of phases and 100 features. Interestingly, one can observe that for very small training set sizes (2% and 5%), using 2 phases leads to significantly better results as for 2% of the training data, the classification accuracy is about 75% while it is round 56% when using 5 or 10 phases. The reason is that the estimation of PH distributions with a larger number of parameters[5] becomes unreliable when there are too few observations. When the

[3] `Splice` is available from the UCI repository.

[4] `DBSubloc` is available at `http://www.bioinfo.tsinghua.edu.cn/dbsubloc.html`.

[5] A PH distribution with p phases has $p^2 + p$ parameters.

training set size becomes greater than 10%, the benefit of additional phases is noticed. The experiments were made using up to 20 phases but it appears that using more than 5 phases does not significantly improve the accuracy for this problem. The right side of Figure 2 shows the accuracy obtained on validation

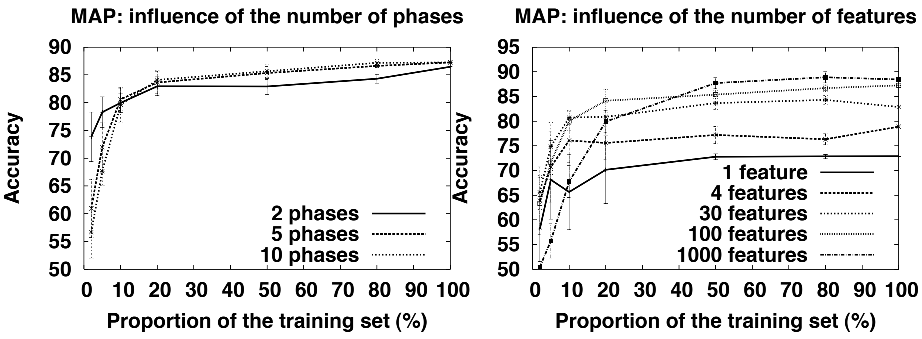


Fig. 2. Influence of the number of phases and of the number of features on the MAP classifier using validation data. Left: influence of the number of phases using 100 features. Right: influence of the number of features using 5 phases.

data using the MAP classifier with an increasing number of features (1, 4, 30, 100 and 1000) and 5 phases. The features were selected according to their JS ranking (see section 4). It can be observed that, in general, the classification accuracy increases with the number of selected features. An accuracy around 73 % is already obtained using a single feature (the highest ranked feature is the substring pair $(\mathtt{T}, \mathtt{GG})$). Using 1000 features only improves the results when 50% of the training data are used as for smaller training set sizes, the lack of observations (for rare features) decreases the accuracy of the fitting. The best accuracy on validation data, 90%, is obtained using the complete training set, 1000 features and 10 phases.

Practical evaluations of SVM with the PH kernel illustrate robustness to overfitting of PH estimations. Indeed, the number of phases has no influence on the results and the method performs well even with small training set sizes and 1000 features. The SVM classifier thus seems able to compensate unreliable estimations by adapting its decision function.

Figure 3 shows comparative results on both dataset using several classifiers: (i) smoothed N-grams [5], (ii) MAP, (iii) SVM with the PH kernel and (iv) SVM with the *blended*[6] *spectrum* string kernel [10] (p. 350). While the PH kernel relies on passage times between substrings, the blended spectrum string kernel is based on the frequencies of all common substrings up to a fixed length. The SVM regularization constant C was tuned according to the heuristic[7] of SVM^{light}

[6] Experiments with the standard p-spectrum kernel [10] (p. 347) offer worse results than the blended version reported here.

[7] $C = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\sqrt{k(s_i,s_i)}}$ where n is the number of training sequences.

and the kernel parameters were selected using the validation set. The left side of Figure 3 presents results obtained on the `Splice dataset`. The best parameters obtained on validation data are a 4-gram, a blended spectrum string kernel length of 7 with a length weight of 3.5, and a 10 phases PH modeling of the 1000 best features for our methods. When a sufficient amount of data is used (at least 50% of the training data), the best performances are obtained with the MAP classifier. Both kernels have comparable performance on this task. The test classification accuracy for the 4-grams, the PH kernel, the blended spectrum string kernel, and the MAP classifiers are respectively 84.1%, 88.5%, 88.8% and 89.7% when the whole training set is used. The right side of Figure 3 presents results obtained

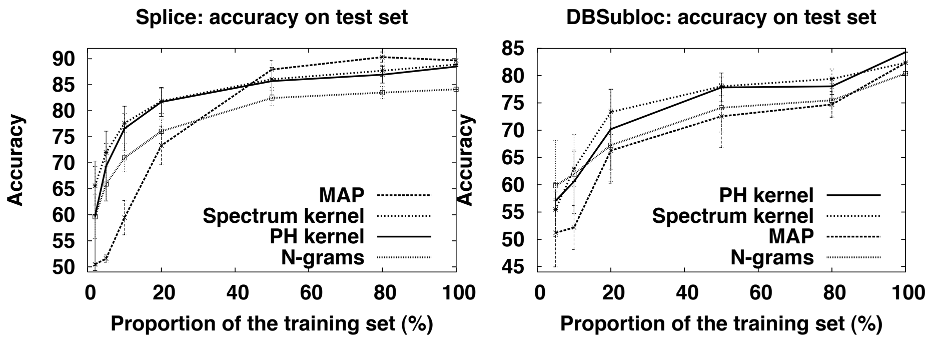


Fig. 3. Comparative classification results for N-grams, MAP and SVM with the PH kernel and a spectrum string kernel on `Splice` (left) and `DBSubloc` (right) test data.

on the `DBSubloc` database. The best parameters obtained on validation data are a 2-gram, a blended spectrum string kernel length of 4 with a length weight of 2.5 and a 5 phases PH modeling of the 100 best features for our methods. The test classification accuracy for 2-grams, the blended spectrum string kernel, the MAP classifier and the PH kernel are respectively 80.4%, 82.35%, 82.4% and 84.3% when the whole training set is used.

6 Conclusion

We propose in this paper a novel approach to the classification of discrete sequences. This approach builds a model fitting some dynamical features deduced from the learning sample. More precisely, the distribution of the times between occurrences of substrings observed in the sample are modeled with discrete phase-type (PH) distributions. Phase-type distributions are defined as the distribution of the time to absorption in finite absorbing Markov chains. This kind of modeling is powerful as it allows one to decompose complex distributions as a combination of phases. The PHit algorithm, an adapted version of the Expectation-Maximization algorithm, is proposed to estimate PH distributions. In this context, tuning the number of phases allows one to deal with the *bias-variance* trade-off. The most informative features (pairs of substrings) are selected according to the Jensen-Shannon divergence between their class conditional empirical FPT distributions. The selected features are used in two classification schemes: a maximum a posteriori (MAP) classifier and support vector machines (SVM) with marginalized kernels. Experiments on DNA splicing region detection and on protein sublocalization illustrate that the proposed techniques offer better results than smoothed Markov chains and competitive results with SVM and a blended spectrum string kernel.

Our future work includes the evaluation of the proposed methods when noisy training data are considered. The HMM learning technique proposed in [3] could also be extended in order to fit first passage time distributions between every pair of symbols, rather than simply the expectations of these times.

Acknowledgment

The authors wish to thank Philippe Delsarte and Guy Latouche for many fruitful discussions about this work. This work is partially supported by the *Fonds pour la formation à la Recherche dans l'Industrie et dans l'Agriculture (F.R.I.A.)* under grant reference F3/5/5-MCF/FC-19271.

References

1. Søren Asmussen, Olle Nerman, and Marita Olsson. Fitting phase-type distributions via the em algorithm. *Scandinavian Journal of Statistics*, 23(4):419–441, 1996.
2. A. Bobbio, A. Horváth, M. Scarpa, and M. Telek. Acyclic discrete phase type distributions: properties and a parameter estimation algorithm. *Perform. Eval.*, 54(1):1–32, 2003.
3. Jérôme Callut and Pierre Dupont. Inducing hidden markov models to model long-term dependencies. In *16th European Conference on Machine Learning (ECML)*, number 3720 in Lecture Notes in Artificial Intelligence, pages 513–521, Porto, Portugal, 2005. Springer Verlag.
4. A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society Ser. B (methodological)*, 39:1–38, 1977.
5. P. Dupont. Noisy sequence classification with smoothed markov chains. In *Conférence francophone sur l'apprentissage automatique 2006, (CAp 2006)*, pages 187–201, Trégastel, France, 2006.
6. J.G. Kemeny and J.L. Snell. *Finite Markov Chains.* Springer-Verlag, 1983.
7. G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling.* Society for Industrial & Applied Mathematics,U.S., 1999.
8. J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Information Theory*, 37:145–151, 1991.
9. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
10. John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, June 2004.