Improved smoothing for Probabilistic Suffix Trees seen as variable order Markov chains

Christopher Kermorvant[1] and Pierre Dupont[2]

[1] EURISE, Université Jean Monnet, Saint-Etienne, France
[2] INGI, University of Louvain, Louvain-la-Neuve, Belgium

Abstract.[1] In this paper, we compare Probabilistic Suffix Trees (PST), recently proposed, to a specific smoothing of Markov chains and show that they both induce the same model, namely a variable order Markov chain. We show a weakness of PST in terms of smoothing and propose to use an enhanced smoothing. We show that the model based on enhanced smoothing outperform the PST while needing less parameters on a protein domain detection task on public databases.

1 Introduction

In many application domains like spoken or written natural language recognition and biological sequences analysis, Markovian models are widely used to model probability distributions on sequences of events. However, in the case of most general models in this family (hidden Markov models), the training and decoding procedures can be computationally heavy. Therefore, in many cases, the use of Markov chains, a subclass of Markovian models, is considered.

Recently, a model based on variable order Markov chains, called probabilistic suffix trees (PST) has been proposed. On a computational biology task (protein domains detection), its performance is competitive with models based on hidden Markov models, while being of lower algorithmic complexity both for its learning procedure and for domain detection [3]. This model can be trained from raw sequences, whereas HMM based models may need aligned sequences, meaning that it is independent from an alignment procedure. This is an important point since multiple alignment of protein sequences is a computer consuming task and is liable to errors. Moreover, the EM algorithm used to train HMM can hit a local minimum.

However, the model based on PST recently proposed provides a very simple solution to a problem pointed out as very important for models based on Markov chains : the probability smoothing. This classical problem occurs in probability estimation when the number of possible events is very large compared to the number of observed events. In this case, many non observed yet possible events

[1] Published in *Proceedings of the 13th European Conference on Machine Learning*, LNAI, No. 2430, Springer Verlag, pp. 185–194, Helsinki, Finland, 2002.

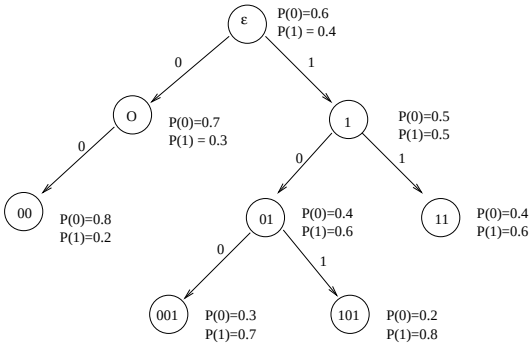

Fig. 1. A Probabilistic Suffix Tree on $\Sigma = \{0, 1\}$. The probability assigned to the sequence 100110 by this tree is $P(100110) = \gamma_e(1)\gamma_1(0)\gamma_0(0)\gamma_{00}(1)\gamma_{001}(1)\gamma_{11}(0) = 0.4 * 0.5 * 0.7 * 0.2 * 0.7 * 0.4 = 7.84\ 10^{-3}$

are estimated with a null probability. Smoothing probability estimation consists in estimating the probability of non observed events, while correcting the probability of observed events.

The structure of this article is as follows. We first recall the definition of probabilistic suffix trees, present the link with Markov chains, give the inference algorithm and point out the weakness of the model regarding probability smoothing. Then, we present a smoothing technique which has been proved to be efficient and show that this technique induces a model which is equivalent to PST but with a better smoothing. Finally, we test the models on a protein detection task and show that using the improved smoothing technique yields better results with less parameters.

2 Variable order Markov chains

2.1 Prediction suffix trees

Probabilistic suffix trees (PST), also known as prediction suffix trees, are a model of probability distribution for discrete events occurring in sequences. They can be used either to predict the next event in the sequence, as in language modeling, or to assign a probability to a whole sequence for classification, or detection as in protein domain detection. The basic assumption underlying the PST is that the probability of an event in the sequence depends at most on the k preceding events, for some k fixed. The difference between PST and classical Markov chains is that for Markov chains, the dependency relies on exactly the k preceding events, whereas in PST the dependency can be based on a number of preceding events varying from 0 to k. In that way, PST are a generalization of Markov chains. Formally, let Σ be a set of discrete symbols (events). A PST is a rooted tree of maximum degree $|\Sigma|$ for which

- for each node, there is at most one outgoing edge labeled by each symbol of Σ

1 **begin**
2 T has only one node, labeled by ϵ (the empty suffix)
3 $S = \{\sigma | \sigma \in \Sigma,\ P(\sigma) \geq P_{min}\}$
4 **while** $S \neq \emptyset$, select s in S
5 remove s from S
6 **if** $\exists \sigma \in \Sigma$ such that $P(\sigma|s) \geq \gamma_{min}$ and $\frac{P(\sigma|s)}{P((\sigma|suf(s))} \leq \frac{1}{r}$ or $\geq r$
7 **then** add to T the node labeled by s and all the nodes needed to go
8 to this node from the node in the tree labeled by the largest suffix of s.
9 **end_if**
10 **if** $|s| < L$ **then** $S = S \cup \{\sigma s \mid \sigma \in \Sigma,\ P(\sigma s) > P_{min}\}$ **end_if**
11 **end_while**
12 **for all** s labeling a node in T, smooth the probability such that
13 $\forall \sigma \in \Sigma,\ \hat{\gamma}_s(\sigma) = (1 - |\Sigma|\gamma_{min})P(\sigma|s) + \gamma_{min}$
14 **end**

Fig. 2. The PST inference algorithm.

- each node is labeled by the labels sequence of the edges needed to go up from this node to the root (suffix labeling).
- for each node q, every outgoing transition on a symbol s is associated to a probability $\gamma_q(s)$ such that $\sum_{s \in \Sigma} \gamma_q(s) = 1$

Let $X = (x_n)_{n \in \{1,\ldots,n\}}$ be a sequence of events. The probability assigned by the PST to the sequence is given by $P(x_1, \cdots, x_n) = \prod_{i=1}^{n} \gamma_{s_i}(x_i)$ where s_i is the node of the tree corresponding to the largest suffix of $x_1, \cdots, x_{i-1}$ stored in the tree. An example of suffix tree is presented on Figure 1.

Ron et *al.* [14] proposed an algorithm for PST inference and studied the learnability properties of the model. Given a target PST with n states and with maximum depth L, and for every given $\epsilon > 0$ and $0 < \delta < 1$, their algorithm returns with confidence $1-\delta$, in time polynomial in n, L, $|\Sigma|$, $\frac{1}{\delta}$, $\frac{1}{\epsilon}$, a PST whose per symbol Kullback-Leibler distance from the target is at most ϵ.

The PST inference algorithm is presented on Figure 2. Given a sequence $s = x_0, \cdots, x_i$, we note $suf(s) = x_1, \ldots, x_i$ the largest suffix of s different from s. The algorithm starts from a 0th order Markov chain (line 2) and with S, the set of suffixes to be examined, containing all the symbols with probability larger than a threshold set to P_{min} (line 3). Then, for each element s in S, if there exists a symbol for which the probability conditioned by s is significantly different from the probability conditioned by the suffix of s (line 4), then the node labeled by s is added to the tree, and also all the nodes needed to go to this node from the node in the tree labeled by the largest suffix of s. If the length of s is strictly smaller than the maximum order of the tree, then all the sequences built from s added with a symbol σ, such that the probability of the sequence $s\sigma$ is larger the threshold P_{min} are added to S (line 10). The last step of the algorithm is a very simple smoothing procedure. The maximum likelihood probability estimator is

modified so that no symbol is predicted with probability 0 whatever its suffix is (line 13).

However, the smoothing procedure using the modified maximum likelihood estimator has two main drawbacks. First, the same constant value γ_{min} is added to every probability, whatever the observed frequency and the probability of the event are. Second, the same floor probability γ_{min} is assigned to all unseen events, whatever their suffix is. Many other smoothing procedures, which do not suffer from these problems, have been proposed. We present one of them in the next section.

2.2 Back-off smoothing of Markov Chains

When using Markovian models in real applications, even if a large amount of data is available to estimate the model, the problem of predicting events which were not observed during the estimation procedure occurs. This is particularly true for high order Markovian models since the number of possible contexts for a kth order Markovian model is exponential in k. The probability of a large number of events are estimated on only a few occurrences, leading to poor probability estimation. Many rare but possible events are not observed, and are wrongly estimated with a null probability.

The problem of predicting unseen events, also known as the zero-frequency problem, is due to the fact that the maximum likelihood estimator attributes the whole probability mass to the events seen during the estimation. Several solutions have been proposed to this problem : succession laws [12], linear interpolation of the maximum likelihood estimator with another estimator, such as an *a priori* distribution [8] or a more general distribution [7], discounting of a certain amount of the probability mass of seen events using the Turing-good formula [9] or absolute discounting [13]. When using discounting, the discounted probability mass is redistributed to all unseen events according to another probability distribution. This is the back-off smoothing methods, proposed by Katz [9].

For Markov chains, the back-off smoothing is based on the following idea : discount a certain amount d_C from the probability mass of events which have been observed in a context of length k and redistribute this amount to all unseen events according to their probability in a context of length $k-1$. This probability can in turn be recursively smoothed. Formally, recalling the notation introduced in the previous section, if σ is a symbol and s a suffix (context), we have :

$$P(\sigma|s) = \begin{cases} \frac{c(s,\sigma)-d_C}{\sum_{\sigma \in \Sigma} c(s,\sigma)} & \text{if } c(s,\sigma) > 0 \\ \alpha(s)\beta(s,\sigma) & \text{otherwise} \end{cases}$$

where $c(s,\sigma)$ is the number of times σ was seen after the suffix s and d_C is the discount parameter, which may depend on $c(s,\sigma)$, $\alpha(s)$ is a normalization factor and $\beta(s,\sigma)$ is the back-off distribution, generally estimated on shorter suffixes. This distribution can in turn be smoothed, inducing a recursive process which ends, in the worst case, with the unconditional probability of the symbol $P(\sigma)$.

In this case, the back-off distribution is used only if the main distribution is null (shadowing). Kneser and Ney [10] showed that using the back-off distribution even if the main distribution is not null (non-shadowing) leads to a better model. We have then :

$$P(\sigma|s) = \begin{cases} \frac{c(s,\sigma)-d_C}{\sum_{\sigma \in \Sigma} c(s,\sigma)} + \alpha'(s)\beta(s,\sigma) & \text{if } c(s,\sigma) > 0 \\ \alpha'(s)\beta(s,\sigma) & \text{otherwise} \end{cases}$$

where $\alpha'(s)$ is a normalization factor. This method is also named non-linear interpolation, since it can be defined as

$$P(\sigma|s) = \max\left[\frac{c(s,\sigma) - d_C}{\sum_{\sigma \in \Sigma} c(s,\sigma)}, 0\right] + \alpha'(s)\beta(s,\sigma)$$

if we suppose that $d_C \leq c(s,\sigma)$ for all $c(s,\sigma) > 0$. The normalization factor is then $\alpha'(s) = \sum_{\sigma \in \Sigma \mid c(s,\sigma)>0} \frac{d_C}{\sum_{\sigma \in \Sigma} c(s,\sigma)}$. Kneser and Ney propose to estimate the back-off probability in the following way :

$$\beta(s,\sigma) = \frac{c(\bullet, suf(s), \sigma)}{\sum_{\sigma \in \Sigma} c(\bullet, suf(s), \sigma))}$$

with $c(\bullet, suf(s), \sigma) = \left|\{\sigma' \mid c(\sigma', suf(s), \sigma) > 0\}\right|$. This estimation is not based on the observed frequency of the sequence (s,σ) but on the number of different contexts in which σ has been observed after $suf(s)$.

The back-off probability can also be null, leading to a recursive smoothing using the same formula. Therefore, we see that using a recursive smoothing on a kth order Markovian model leads to build a variable order Markov chain. There are three differences between a variable order Markov chain build by recursive Kneser-Ney back-off smoothing (denoted KN-chain) and a variable order Markov chain represented by a PST infered by Ron's algorithm:

- there is no pruning in KN-chain : if a sequence of length lower than the maximum order of the KN-chain is observed, its probability is estimated and stored, whereas in PST the estimated probability is stored only if it is above a threshold (P_{min}). For a given maximum order, PST may have less parameters than KN-chains.
- in KN-chain, a different and enhanced estimation scheme is used for the back-off probability estimation, whereas in PST, for all the orders, probability estimation is based on the modified maximum likelihood estimator.
- in KN-chain, both the enhanced estimation scheme and the modified maximum likelihood estimator are used (non-shadowing), whereas in PST, only one modified maximum likelihood estimator is used.

In the next section, we show that KN-chains significantly outperform PST on a protein domain detection task.

3 Application to protein domains detection

Many databases have been created to gather information concerning proteins. Researchers can find in these databases not only the amino-acid sequence of proteins but also information about their functions, structure, related diseases and bibliographical pointers. These databases are used to help the analysis of newly sequenced proteins, for which no function or structure is known yet. They serve as a basis for learning models which are used to detect sub-sequences (called domains or motifs) which are known to be related to a particular biochemical function. Such models range from complex probabilistic models based on hidden Markov models [11, 5] to purely syntactic models, like regular expressions, describing characteristic sub-sequences [1]. However, since the databases are constantly increasing and updated, the learning procedure of these models must be easy and of low complexity.

3.1 Protein domains detection with variable Markov chains

Automatic analysis of newly sequenced proteins, for which neither structure nor biochemical functions are known yet, is now very important since the number of newly sequenced proteins is increasing daily. To a certain extend, hypotheses concerning the function of a protein can be made by searching, in its amino-acid sequence, sub-sequences which are known to be related to a function in other proteins. Many of such sub-sequences, called domains, have been identified and are stored in databases like PFAM [15]. However, the sequence of a given domain is not constant through species. Substitutions, deletions and insertions occur, which make domain detection more complex than a simple exact sub-sequence detection.

Domain models, like HMM[5], are trained on theses sub-sequences and used to detect domains in complete protein sequences. Variable order Markov chains may also be used to detect domains in protein sequences [3]. A variable order Markov chain is associated to each domain to be detected and is estimated on a set of examples of such domain. Then the likelihood of a new protein sequence given a domain model is related to the presence or not of the corresponding domain in the protein. A high likelihood is a sign of a probable presence of the corresponding domain in the protein.

3.2 Experimental setup

We used two databases to test our models : the SWISSPROT database [2] which contains protein sequences from several living organisms and the PFAM database [15] which contains alignments of functional domains, grouped in families, extracted from SWISSPROT with a semi-automatic procedure. We labeled the SWISSPROT sequences with the name of the domains they contain, according to PFAM families. In order to compare with recently publish results [6, 3], we used PFAM release 1.0. This release contains 22307 domains grouped in 175 families.

	Kneser-Ney Smoothing					PST
Maximal order	0	1	2	3	4	20
Correct detection rate	13.9	53.0	81.3	89.5	90.0	85.8
Number of parameters	$2,2 \times 10^1$	$4,0 \times 10^2$	$3,3 \times 10^3$	$9,9 \times 10^3$	$1,8 \times 10^4$	$5,1 \times 10^4$

Fig. 3. Correct detection rate on the *complete* SWISSPROT database and number of parameters for variable order Markov chains with Kneser-Ney smoothing and PST.

3.3 Training the models

For each domain family, we estimated the models on 80% of the domain sequences extracted from the alignments available in PFAM. We trained probabilistic suffix tree with software and parameters given as optimal by Bejerano [3]. The maximal order of the PST is 20. We also trained variable order Markov chains with Kneser-Ney smoothing, with maximum order ranging from 0 to 4.

3.4 Testing the models

All the models were tested for domain detection on the protein sequences of the SWISSPROT database corresponding to the complete PFAM database. In order to measure a correct detection rate, we used the iso-point detection criterion [3, 6]. For each family model, an iso-point is computed on the complete SWISSPROT sequences set. The iso-point is defined as the value v for which the number of protein sequences not containing the domain with a likelihood above v is equal to the number of protein sequences containing the domain with a likelihood under v. For a given model, a sequence containing the domain with a likelihood above the iso-point is considered correctly detected. The correct detection rate is defined as the ratio of the number of proteins correctly detected on the number of proteins containing the domain. Note that in order to compute the iso-point, the likelihood of each sequence is normalized by its length.

3.5 Results

Figure 3 shows the correct detection rate on all the SWISSPROT database sequences and the number of parameters for PST and variable order Markov chains with Kneser-Ney smoothing. Smoothed variable order Markov chains outperform PST as soon as the maximum order of the chains is greater or equal to 3. As from the 4th order Markov chain and up to the the 9th order Markov chain, the detection rate is stationary. Considering the domain detection problem as a binary classification problem ("does a sequence contain a given domain or not"), the performance difference between variable 4th order Markov chains and PST was tested with a McNemar test [4]. The $H0$ hypothesis "the variable 4th order Markov chains and PST have the same classification performance" was rejected (p-value$< 10^{-15}$). The performance difference is thus significant.

Figure 4 shows the detection rate on the part of the SWISSPROT database corresponding to the PFAM domains which were *not* used for training (named

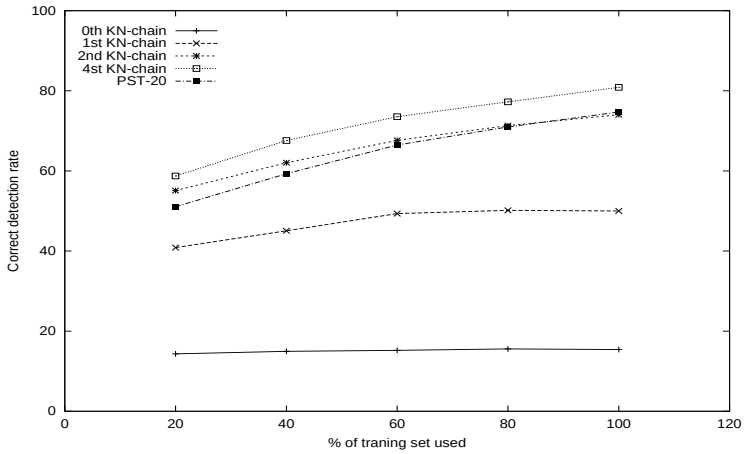

Fig. 4. Learning curve: correct detection rate on the SWISSPROT test set versus the size of the PFAM training set for Markov chains with Kneser-Ney smoothing with order ranging from 0 to 4 (*0th KN-chain to 4th KN-chain*) and PST with maximum order 20 (*PST20*)

SWISSPROT test set). Results are given when the size of the training set is varying from 20% to 100%. Even on small training set, the 4th-order smoothed variable order Markov chains outperform PST.

Finally, figure 5 shows the detection rate with respect to the number of parameter needed by the model. The 3rd-order and 4th-order Markov chains outperform PST while needing significantly less parameters.

4 Conclusion

We have shown that PST and smoothed Markov chains can be seen as equivalent variable order Markov models but for the smoothing technique. As the quality of the back-off technique as been shown to be important for Markov chain in other application domains, we proposed to enhance the smoothing technique used in PST by using a non-shadowing back-off smoothing to lower order Markov chain estimated as proposed by Kneser and Ney [10]. With this improved smoothing, we showed that the maximum order of the Markov chain can be drastically reduced, with a performance increase on a protein domain detection task. By reducing the maximum order of the Markov chain, we also reduce the number of parameters needed.

References

1. A. Bairoch. PROSITE: A dictionary of sites and patterns in proteins. *Nucleic Acids Research*, 19:2241–2245, 1991.

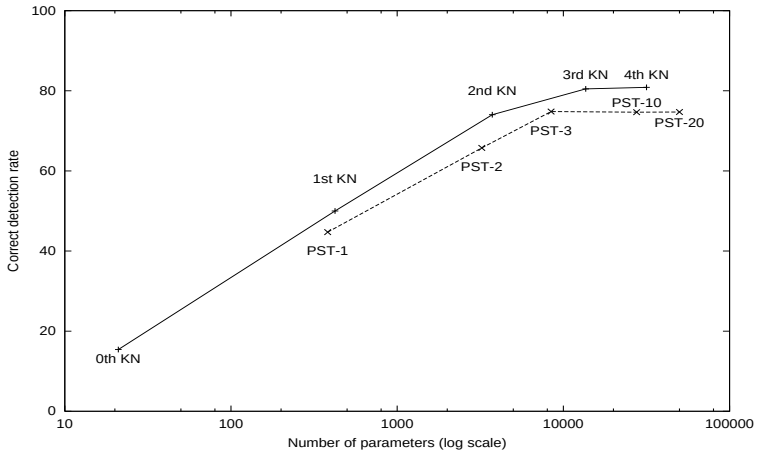

Fig. 5. Detection rate on the SWISSPROT test set versus the number of parameters needed for Markov chains with Kneser-Ney smoothing with order ranging from 0 to 4 (*0th KN-chain to 4th KN-chain*) and PST with maximum order ranging from 1 to 20 (*PST1 to PST20*)

2. A. Bairoch and R. Apweiler. The SWISS-PROT protein sequence data bank and its new supplement TrEMBL. *Nucleic Acids Res.*, 24:21–25, 1996.
3. Gill Bejerano and Golan Yona. Variations on probabilistic suffix trees: statistical modeling and prediction of protein families. *Bioinformatics*, 17(1):23–43, Jan 2001.
4. T. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18:97–136, 1997.
5. S. Eddy. *HMMER user's guide: biological analysis using profile hidden Markov models.* Department of Genetics, Washington University School of Medecine, 1998. http://hmmer.wustl.edu/.
6. E. Eskin, W. Grundy, and Y. Singer. Protein family classification using sparse markov transducers. In *Proc. Int. Conf. on Intelligent Systems for Molecular Biology*, August 2000.
7. F. Jelinek and R. Mercer. Interpolated estimation of markov source parameters from sparse data. In E. Gelsema and L. Kanal, editors, *Pattern recognition in practice*, pages 381–397, Amsterdam, 1980. North-Holland.
8. W.E. Johnson. Probability : deductive and inductive problems. *Mind*, 41:421–423, 1932.
9. Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, March 1987.
10. R. Kneser and H. Ney. Improved backing-off for M-gram language modeling. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 181–184, Detroit, MI, May 1995.
11. A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.
12. G. Lidstone. Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities. *Trans. Fac. Actuar.*, 8:182–192, 1920.

13. Hermann Ney, Ute Essen, and Reinhard Kneser. On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38, 1994.
14. D. Ron, Y. Singer, and N. Tishby. The power of amnesia. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 176–183. Morgan Kaufmann Publishers, Inc., 1994.
15. E. L. L. Sonnhammer, S. R. Eddy, and R. Durbin. Pfam : a comprehensive database of protein domain families based on seed alignments. *Proteins*, 28(3):405–420, 1997.