ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

POKer: a Partial Order Kernel for Comparing Strings with Alternative Substrings

Maryam Abdollahyan and Fabrizio Smeraldi

School of Electronic Engineering and Computer Science
Queen Mary University of London, Mile End Road, London E1 4NS - UK

Abstract. We introduce a Partial Order Kernel (POKer) on the weighted sum of local alignment scores that can be used for comparison and classification of strings containing alternative substrings of variable length. POKer is defined over the product of two directed acyclic graphs, each representing a string with alternative substrings, and is computed efficiently using dynamic programming. We evaluate the performance of POKer with Support Vector Machines on a dataset of strings generated by detecting overlapping motifs in a set of simulated DNA sequences. Compared to a generalization of a state-of-the-art string kernel, POKer achieves a higher classification accuracy.

1 Introduction

String comparison is a fundamental operation in computer science, with applications in many areas including text retrieval, data compression and computational biology. Particularly in the latter application, it is often necessary to handle the uncertainty in the strings to be compared. Sequence alignment algorithms based on dynamic programming are very effective in dealing with uncertainty involving single character substitutions (as in "hello" and "hullo") [8, 10]; recently developed variations of these algorithms allow the comparison of strings containing alternative substrings that can be substituted for each other (henceforth, strings with alternatives). This is done by representing alternative substrings (e.g. "man" and "person" in "spokes[man|person]ship") as paths in a directed acyclic graph (DAG). Such a graph can then be aligned to another DAG (e.g. representing "spokes[woman|people]") to provide a measure of similarity between two strings based on an alignment of best matches (likely "spokesmanship" and "spokeswoman") [2, 4, 5]. The resulting alignment score, however, is in general not a metric, making it impractical for use with many learning algorithms. This motivated us to develop a kernel for strings with alternatives.

We present a convolution kernel for the comparison of two DAGs, each representing a partial order over the characters of a string with alternatives. Our Partial Order Kernel (POKer) uses dynamic programming to efficiently compute an exponentially weighted sum of the local alignment scores corresponding to all choices of paths in the two graphs. In addition to being able to handle alternative substrings of different lengths, POKer (which can be seen as a generalization of the LA kernel [11] and a member of the family of rational kernels [3]) also provides a metric interpretation of sums of partial order alignment scores [5] in

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

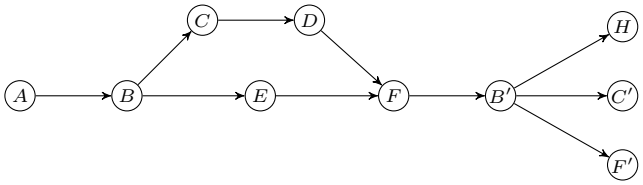

Fig. 1: DAG representation of $AB[CD|E]FB'[H|C'|F']$

the form of the norm induced by the kernel. This can be used in combination with other learning algorithms beyond kernel methods.

We evaluate POKer on a dataset obtained by detecting motifs in simulated DNA sequences, a common task in computational biology that, due to the overlapping of motifs, naturally leads to a collection of strings with alternatives. We use POKer in conjunction with SVMs to classify this dataset and show that compared to a generalization of the popular spectrum kernel [7], POKer has a better discrimination ability.

2 DAG Representation of Strings

Let x be a string with alternative substrings. For example, consider the string $x = AB[CD|E]FB'[H|C'|F']$ where brackets indicate that, for instance, substring CD appears in x as an alternative to character E (we use a prime for convenience to distinguish between multiple occurrences of the same character). We denote the multiset of characters in x by $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ and its set of standard strings, generated by all possible choices of alternatives, by Alt_x (e.g. $ABEFB'C' \in Alt_x$). Note that repeated characters count as distinct elements in $\mathcal{X}$. For any two characters $x_i, x_j \in \mathcal{X}$ we write $x_i \prec x_j$ if x_i precedes x_j in some string in Alt_x. The relation $\prec$ is a partial order on $\mathcal{X}$ (as for instance, $A \prec E$ and $C \prec D$ but neither $H \prec C'$ nor $C' \prec H$).

We represent $(\mathcal{X}, \prec)$ as a directed acyclic graph (DAG) G_x, where $V(G_x) = \mathcal{X}$ is the set of vertices and an edge exists between x_i and x_j if and only if $x_i \prec x_j$. Each path in G_x, from a source to a sink vertex, corresponds to a string in Alt_x (there may exist multiple source/sink vertices since x can start/end with alternatives). The DAG representation of the above example string is shown in Figure 1.

3 Partial Order Kernel

3.1 Definition

Let x and y be two strings with alternative substrings. The Partial Order Kernel (POKer) $K(x, y)$ is defined as a convolution of (several copies of) two kernels K_a, K_g with respect to relations R_x^m and R_y^m over the vertices of paths in G_x and G_y, respectively (see [6] for a definition of convolution kernels). We define

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

R_x^m as the set of lists $(X_1, X_2, \ldots, X_{2m-1})$, $X_i \subseteq V(G_x)$ such that

1. There exists a path π_x in G_x such that $X_i \subseteq V(\pi_x)\, \forall i$ and $\cup_i X_i = V(\pi_x)$;
2. the X_i are consecutive, i.e. for all $v \in X_i$ and $u \in X_{i+1}$ we have $v \prec u$.

Note that the X_i are disjoint, and that some of them may be empty. R_y^m is defined in a similar way. Intuitively, R_x^m represents a local alignment of m characters along π_x, possibly separated by gaps, with m characters along a path π_y in G_y. This intuition guides the definition of the two kernels to be convolved, namely the substitution kernel K_a and the gap kernel K_g. Let $X \subseteq V(\pi_x)$ and $Y \subseteq V(\pi_y)$. Similarity of aligned characters is measured by the substitution kernel

$$K_a{}^{(\beta)}(X,Y) = \begin{cases} exp(\beta s(X,Y)) & \text{if } |X| = 1 \text{ and } |Y| = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $\beta \geq 0$ is a parameter and $s(X,Y)$ is the substitution score for the labels of vertices in X and Y (specified by e.g. a scoring matrix). Valid values for β are those for which the kernel remains positive semi-definite. Penalty for gaps is quantified by the gap kernel

$$K_g{}^{(\beta)}(X,Y) = \exp\left(\beta g \left(|X| + |Y|\right)\right) \tag{2}$$

where g is a (linear) gap penalty. We convolve the above kernels to construct the K_m kernel

$$K_m(x,y) = (K_a * K_g)^{m-1} * K_a = \sum_{X \in R_x^m, Y \in R_y^m} \left[\prod_{k=1}^{m-1} K_a{}^{(\beta)}(X_{2k-1}, Y_{2k-1}) K_g{}^{(\beta)}(X_{2k}, Y_{2k}) \right] K_a{}^{(\beta)}(X_{2m-1}, Y_{2m-1}) \tag{3}$$

It is clear from the definition of K_a that the terms in the above sum are zero unless all the X_i and Y_i with odd indexes are singletons, and that

$$K_m(x,y) = \sum_{X \in R_x^m, Y \in R_y^m} \exp(\beta S(X,Y)) \tag{4}$$

where $S(X,Y)$ is the score of the local alignment of m characters along π_x with m characters along π_y specified by (X,Y). Note that K_m is zero when m is larger than the length of the longest path in G_x or in G_y. With the above definitions, we now define the POKer

$$K(x,y) = \sum_{m \geq 0} K_m(x,y) = \sum_{m \geq 0} \sum_{X \in R_x^m, Y \in R_y^m} \exp(\beta S(X,Y)) \tag{5}$$

POKer is equal to an exponentially weighted sum of the scores of all the local alignments between any number of characters in x and the same number of characters in y, selected from any paths in G_x and G_y, that is, from any choice of alternatives. In the next section, we show how this (finite) sum can be computed efficiently using dynamic programming.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

3.2 Computation

We will assume, without loss of generality, that both x and y begin with a start character $x_0 = y_0 = \phi$, that is, G_x and G_y each have a single source vertex labelled ϕ, and $s(\phi, \phi) = s(x_i, \phi) = s(\phi, y_j) = 0$ for all i and j. POKer is computed efficiently using dynamic programming over the strong product graph $G_{xy} = G_x \boxtimes G_y$, with a complexity that is linear in the number of vertices of G_{xy}, here indicated simply as (i, j):

$$K(x, y) = 1 + \sum_{(i,j) \in V(G_{xy})} M(i, j) \tag{6}$$

where $M(i, j)$ is computed recursively based on the following

$$\begin{cases} M(i, \phi) = M(\phi, j) = 0 \\ N(i, \phi) = N(\phi, j) = 0 \end{cases} \tag{7}$$

and

$$\begin{cases} M(i,j) = & \exp(\beta s(i,j))[1 + \sum\limits_{mi \in E(G_x), nj \in E(G_y)} N(m,n)] \\ N(i,j) = & \exp(\beta g) \sum\limits_{mi \in E(G_x)} N(m,j) \\ & + \exp(\beta g) \sum\limits_{nj \in E(G_y)} N(i,n) \\ & - \exp(2\beta g) \sum\limits_{mi \in E(G_x), nj \in E(G_y)} N(m,n) \\ & + M(i,j). \end{cases} \tag{8}$$

where $\beta \geq 0$ is a parameter and g is the gap penalty.

Each local alignment corresponds to a path in G_{xy}. POKer (Equation 6) is a sum over the exponentiated scores of all the local alignments ending at each vertex (i, j) in G_{xy} (including the empty alignment). The contributions of all the local alignments ending at (i, j), including those with the labels of i and j being the only aligned characters is accounted for by $M(i, j)$. Penalties for inserting gaps in x or y at the end of a partial alignment are included in $N(i, j)$. A gap in x followed by a gap in y, and a gap in y followed by a gap in x are equivalent in terms of aligned characters; this is accounted for by the negative term in Equation 8.

4 Experimental Results

We tested POKer with SVMs in a multi-class classification scenario and compared its performance for different number of classes to that obtained by SVMs using a generalized spectrum kernel [7].

We used a standard publicly available tool (rMotifGen [9]) to generate the dataset as follows. First, we produced a dictionary consisting of 50 randomly generated motifs, i.e. short DNA sequences, each representing a binding site.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

We assigned a unique character to each motif. To generate the strings in each class, we randomly chose 10 motifs from the dictionary. We then generated 200 random DNA sequences containing these motifs. We set the parameters so that each motif (out of 10) appears in 70% of the sequences, and used the default values for the rest of the parameters. Finally, we scanned each sequence for the occurrences of all 50 motifs (as motifs others than those explicitly inserted in a sequence can appear in it). This yielded a string of characters corresponding to the detected motifs. Since two or more motifs can overlap but only one protein at a time can bind to the overlapping sites, this string is a string with alternative substrings (similar to Figure 1) [1]. We repeated this procedure 40 times to generate 40 classes of 200 strings with alternatives, i.e. a total of 8000 strings.

To the best of our knowledge, no other kernels have been specifically designed for strings with alternative substrings of variable length. However, the popular spectrum kernel [7] can be extended in order to deal with such strings and provide a baseline for comparison. We define this generalized kernel as follows. Let x be a string with alternatives over an alphabet $\mathcal{A}$, and G_x its DAG representation. We denote the set of all paths of length k in G_x, i.e. k-mers in x, by $P_{(k,x)}$. We define the feature map indexed by the set of all k-length substrings α from $\mathcal{A}^k$ as

$$\Phi_k(x) = \left(\left|\{\pi \in P_{(k,x)} | (V(\pi)) = \alpha\}\right|\right)_{\alpha \in \mathcal{A}^k} \tag{9}$$

where $(V(\pi))$ is the sequence of labels of vertices of π. The generalized spectrum kernel is then equal to

$$K_k(x, y) = \langle \Phi_k(x), \Phi_k(y) \rangle \tag{10}$$

Note that this is not equivalent to accumulating the occurrence of each k-mer over all the strings in Alt_x, since substrings common to multiple strings in Alt_x would then be counted more than once.

We built an SVM classifier for each class using the one-versus-all strategy. The match score, mismatch score and gap penalty were set to 4, 0 and -2, respectively. We tested POKer with several β values, ranging from 0 to $+\infty$, and chose the one that yielded the best performance (β=0.1). Similarly, we tested the generalized spectrum kernel with several $k \in \{2, 3, 4, 5\}$ values and chose the one for which the kernel performed best (k=3). We use ROC scores (area under the ROC curve), averaged over the number of classes, to compare the performance of two kernels. The results are shown in Table 1. As can be seen, POKer outperforms the generalized spectrum kernel in all cases, its performance decreasing only marginally as the number of classes increases. Moreover, POKer is fairly robust to the choice of β; in the case of 40 classes, the best mean ROC score is 96.4% when β=0.1 while β=0.01 and β=1 yield 95.8% and 91.2%, respectively.

5 Conclusions

In this work, we presented POKer, a convolution kernel on the weighted sum of local alignment scores for comparing strings containing alternative substrings.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

	Mean ROC score	
No. of classes	**POKer**	**Generalized spectrum kernel**
5	98.4%	93.4%
10	98.1%	90.1%
20	97.8%	86.4%
40	96.4%	82.0%

Table 1: Mean ROC scores obtained by POKer and the generalized spectrum kernel for different number of classes

POKer can handle alternative substrings of different length, is computed efficiently via dynamic programming and has an intuitive interpretation in terms of the sum of local alignment scores for all alternatives. In our experiments we found POKer to be very effective in discriminating between classes of strings with alternative substrings, compared to a generalization of the popular spectrum kernel. With richer mathematical properties than (non-metric) alignment scores, low computational complexity and classification effectiveness, POKer can be a powerful tool in the analysis of strings with alternatives.

References

[1] M. Abdollahyan, F. Smeraldi, B. Noyvert, and G. Elgar, Transcription factor binding site-based alignment of conserved non-coding elements. Poster presented at the European Conference on Computational Biology (ECCB 2016). doi:10.7490/f1000research.1113029.1

[2] L. Arvestad, Algorithms for biological sequence alignment. 1999.

[3] C. Cortes, P. Haffner and M. Mohri, Rational kernels: Theory and algorithms. *Journal of Machine Learning Research*, 5:1035–1062, 2004.

[4] M. Gîrdea, L. Noé and G. Kucherov, Back-translation for discovering distant protein homologies. *Algorithms in Bioinformatics*, 108–120, Springer, 2009.

[5] C. Grasso and C. Lee, Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems. *Bioinformatics*, 20(10):1546–1556, 2004.

[6] D. Haussler, Convolution kernels on discrete structures. Technical report, Department of Computer Science, University of California at Santa Cruz, 1999.

[7] C. S. Leslie, E. Eskin and W. S. Noble, The spectrum kernel: A string kernel for SVM protein classification. *Pacific symposium on biocomputing*. Vol. 7, No. 7, pages 566–575, 2002.

[8] S. B. Needleman and C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[9] E. C. Rouchka and C. T. Hardin, rMotifGen: random motif generator for DNA and protein sequences. *BMC bioinformatics*, 8(1), 2007.

[10] T. F. Smith and M. S. Waterman, Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[11] J. P. Vert, H. Saigo and T. Akutsu, Convolution and local alignment kernels. *Kernel methods in computational biology*, pages 131–154, MIT press, 2004.