Dynamic Alignment Kernels

Chris Watkins

Department of Computer Science
Royal Holloway, University of London
C.Watkins@dcs.rhbnc.ac.uk

Technical Report

CSD-TR-98-11

15th January 1999

Department of Computer Science
Egham, Surrey TW20 0EX, England

Abstract

There is much current interest in kernel methods for classification, regression, PCA, and other linear methods of data analysis. Kernel methods may be particularly valuable for problems in which the input data is not readily described by explicit feature vectors. One such problem is where input data consists of symbol-sequences of different lengths, and the relationships between sequences are best captured by dynamic alignment scores.

This paper shows that the scores produced by certain dynamic alignment algorithms for sequences are in fact valid kernel functions. This is proved by expressing the alignment scores explicitly as dot-products.

Alignment kernels are potentially applicable to biological sequence data, speech data, and time series data.

The kernel construction may be extended from pair HMMs to pair probabilistic context-free grammars.

1 Introduction: Linear Methods using Kernel Functions

In many types of machine learning, the learner is given a training set of *cases* or *examples* $a_1 \ldots a_l \in \mathcal{A}$. $\mathcal{A}$ denotes the set of all possible cases: cases may be vectors, pieces of text, biological sequences, sentences, etc. For supervised learning, the cases are accompanied by a set of corresponding *labels* or *values* $y_1 \ldots y_l$. The cases are mapped to *feature vectors* $x_1 \ldots x_l \in \mathcal{X}$, where the $\mathcal{X}$ is a real vector space termed the *feature space*. The mapping from $\mathcal{A}$ to $\mathcal{X}$ is denoted by ϕ, so that $x_i = \phi(a_i)$. Sometimes the cases are given as feature vectors to start with, in which case ϕ may be the identity mapping; otherwise ϕ denotes the method of assigning numeric feature values to a case.

Once a feature vector x_i has been defined for each case a_i, it becomes possible to apply a wide range of linear methods such as support-vector machines, linear regression, principal components analysis (PCA), and k-means cluster analysis.

As shown in [Vap95] for SV machines, in for example [Wah90] for linear regression, and in [SSM98] for PCA and k-means cluster analysis, the calculations for all of these linear methods may be carried out using a dual rather than a primal formulation of the problem.

For example, in linear least-squares regression the primal formulation is to find a coefficient vector β that minimises $||X\beta - y||$ where X is the design matrix, an l by d matrix in which the ith row is x_i, and each x_i has d elements. If l is larger than d, the usual method of finding β is to solve the normal equations $X^T X\beta = X^T y$. This requires the solution of a set of linear equations with coefficients given by the $d \times d$ matrix $X^T X$.

The dual formulation is to find a coefficient vector α that minimises $||XX^T\alpha - y||$, so that one coefficient α_i is found for each case vector x_i. This requires the solution of a set of linear equations with coefficients given by the $l \times l$ matrix XX^T.

Both methods lead to the same predicted value $\hat{y}$ for a new case x. If there are more cases than features, that is if $l > d$, the primal method is more economical because the $d \times d$ matrix $X^T X$ is smaller than the $l \times l$ matrix XX^T. For example, if there are 200 cases, each described by a vector of 10 measurements, then the primal method requires solving a 10 by 10 system of linear equations, while the dual method requires solving a 200 by 200 system, which will have rank at most 10. For such a problem, the dual method has no advantage.

The potential advantage of the dual method for regression is that it can be applied to very large feature vectors. The coefficient matrix XX^T contains the dot-products of pairs of feature vectors: the ijth element of XX^T is $x_i \cdot x_j$. In the dual calculation, it is only dot-products of feature vectors that are used—feature vectors never appear on their own.

As the feature vectors $x_i = \phi(a_i)$ appear only in dot-products, it is often possible to avoid computing the feature vectors, and to compute dot-products directly in some economical fashion from the case descriptions a_i instead. A *kernel* is a function k that computes a dot-product of feature vectors from the corresponding cases.

Definition 1 *A* kernel *is a function k such that for all $a, b \in \mathcal{A}$,*

$$k(a, b) = \phi(a) \cdot \phi(b)$$

where ϕ is a mapping from $\mathcal{A}$ to a feature space $\mathcal{X}$.

The mapping ϕ determines k uniquely, but k determines only the metric properties of the image under ϕ of the case-set $\mathcal{A}$ in feature space. ϕ is not in general invertible, and indeed $\phi(\mathcal{A})$ need not even be a linear subspace of $\mathcal{X}$. ϕ need not be and in general is not a linear mapping: indeed, addition and multiplication need not even be defined for elements of $\mathcal{A}$.

The dual formulation often has a computational advantage over the primal formulation if the kernel function k is easy to compute, but the mapping to feature space ϕ is infeasible to compute. A well-known example of this is the "homogeneous polynomial kernel" of [Vap95] in which the cases are real d dimensional vectors $a = \langle a^1 \dots a^d \rangle$, and

$$\begin{aligned} k(a, b) &= (a \cdot b)^n && (1) \\ &= \sum_{i_1=1}^{d} \cdots \sum_{i_n=1}^{d} \left(a^{i_1} \cdots a^{i_n}\right) \left(b^{i_1} \cdots b^{i_n}\right) && (2) \end{aligned}$$

for some positive integer n, and $1 \leq i_1, \dots, i_n \leq d$. A mapping ϕ that induces this kernel is

$$\phi(a) = \left\langle a^{i_1} \cdots a^{i_n} : 1 \leq i_1 \dots i_n \leq d \right\rangle \tag{3}$$

In the character recognition application described in [Vap95], the cases were vectors with dimension 256 and values of n up to 8 were used, so that the vectors in (3) had billions of terms, and the expression (1) was vastly easier to compute than the explicit dot-product (2).

2 Applying Linear Methods to Structured Objects

Not all data comes naturally as vectors: data may consist of "structured objects", such as sequences of different lengths, trees, or sentences. To apply linear methods to such data, it is necessary either to construct feature vectors explicitly, or to use a kernel function. The recent success of the methods of [Joa97] in text classification has shown how valuable it can be to apply linear statistical methods to inductive problems where such methods have not previously been used. This section describes three approaches to mapping structured objects to vectors in order to apply linear statistical methods.

2.1 Sparse Vector Kernels

[Joa97] considered the problem of classifying text news stories by subject. Essentially, Joachims considered a text as a sparse vector, with one dimension for each possible word. With an efficient sparse representation, the dot-product of two sparse vectors can be computed in a time proportional to the total number of non-zero elements in the two vectors. A kernel implemented as a sparse dot-product is a natural method of applying linear methods to sequences. Examples of such sparse-vector mappings are:

- ◇ mapping a text to the set of words it contains
- ◇ mapping a text to the set of pairs of words that are in the same sentence
- ◇ mapping a symbol sequence to the set of all subsections of some fixed length m

"Sparse-vector kernels" are an important extension of the range of applicability of linear methods.

2.2 Case-based Features

Often, there are natural matching functions or similarity scores that may be applied to structured objects. These are functions that can be applied to a pair of objects, and which return a real-valued score. Although such a matching is not necessarily representable as a dot-product, any such function can be used to create features in the following way.

Given *any* function $f : \mathcal{A} \times \mathcal{A} \longmapsto \mathbb{R}$, and an indexed set of cases, $a_1, \ldots, a_n$ a possible feature space mapping is

$$\phi(b) = \langle f(a_1, b), \ldots, f(a_n, b) \rangle \tag{4}$$

This is not really a kernel method, as the feature vector is computed explicitly, and there is no computational advantage in using a kernel.

2.3 Diagonal-dominance Kernels

A second canonical construction for a kernel k given any $f : \mathcal{A} \times \mathcal{A} \longmapsto \mathbb{R}$, for a finite or countable set $\mathcal{A}$, uses a feature space with dimensions indexed by $\mathcal{A} \times \mathcal{A}$, and for any $x \in \mathcal{A}$ the $\langle a, b \rangle$th element of the vector $\phi(x)$ is defined as

$$[\phi(x)]_{\langle a,b \rangle} = \begin{cases} f(a,b) & \text{if } a = x \text{ or } b = x \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

so that k is defined as

$$k(a,b) = f(a,b)^2 + f(b,a)^2 \ \text{ if } a \neq b \tag{6}$$

and

$$k(a,a) = f(a,a)^2 + \sum_{c \in \mathcal{A}, c \neq a} f(a,c)^2 + \sum_{c \in \mathcal{A}, c \neq a} f(c,a)^2 \tag{7}$$

This "diagonal-dominance" kernel does in some sense provide a computational advantage, for it enables an arbitrary non-negative symmetric function $k(x,z) = f(x,z)^2 + f(z,x)^2$ for $x \neq z$ to be used as a kernel, provided that the diagonal entries $k(x,x)$ are made sufficiently large that any finite matrix of dot-products of distinct elements of $\mathcal{A}$ will be diagonally dominant, and therefore positive semidefinite.

The size of diagonal element required may be reduced by defining ϕ with respect to a reference data set $\mathcal{R} \subset \mathcal{A}$

$$[\phi(x)]_{\langle a,b\rangle} = \begin{cases} f(a,b) & \text{if } (a = x \text{ or } b = x) \text{ and} \\ & \quad (a \in \mathcal{R} \text{ or } b \in \mathcal{R}\ ) \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

If $\mathcal{R}$ is taken to be a small subset of $\mathcal{A}$—perhaps the training data set itself—then the diagonal elements of the matrix of dot-products of the training data can be set to the sums of the rows. The diagonal elements from (8) may be much smaller than those from (5).

It is curious that this construction of an explicit dot-product for a diagonally dominant matrix only works for matrices with non-negative elements.

Unfortunately matrices with large diagonal elements are likely to provide poor generalisation in learning. Nevertheless, this construction may sometimes be of use.

3 Conditional Symmetric Independence Kernels

Joint probability distributions are often used as scoring functions for matching: two objects "match" if they are in some sense similar, and the degree of similarity or relatedness is defined according to a joint probability distribution that assigns pairs of related objects higher probabilities than pairs of unrelated objects. A joint p.d. used in this way will be described in section (4) below. It is sometimes possible to show that such a joint p.d. is a valid kernel by showing that the p.d. is conditionally symmetrically independent.

Definition 2 *A joint probability distribution is* conditionally symmetrically independent *(CSI) if it is a mixture of a finite or countable number of symmetric independent distributions.*

CSI joint probability distributions may be written as dot-products in the following way. Let X, Z be two discrete random variables, and let p be the joint distribution function, defined as

$$p(x,z) = Pr(X = x \text{ and } Z = z) \tag{9}$$

and let p be symmetric—that is, $p(x,z) = p(z,x)$ for all x, z.

Let C be a random variable such that

$$Pr(X, Z \mid C) = Pr(X \mid C) Pr(Z \mid C) \tag{10}$$

and, given C, the distributions of X and Z are identical. Then

$$p(x, z \mid c) = p(x \mid c) p(z \mid c) \tag{11}$$

for each c in the range $\mathcal{C}$ of C ($\mathcal{C}$ is the set of values that C may take). Then

$$\begin{aligned} p(x,z) &= \sum_c p(x \mid c)p(z \mid c)p(c) \\ &= \sum_c \left(p(x \mid c)\sqrt{p(c)}\right)\left(p(z \mid c)\sqrt{p(c)}\right) \end{aligned} \tag{12}$$

where c takes all values in the range of C. This is a dot-product, with the feature-space mapping defined as

$$\phi(x) = \left\langle p(x \mid c)\sqrt{p(c)} \quad : \quad c \in \mathcal{C} \right\rangle \tag{13}$$

so that

$$p(x,z) = \phi(x) \cdot \phi(z) \tag{14}$$

We believe that this definition can be extended to benign cases in which p is a probability density which is a mixture of an uncountable number of symmetric independent densities, indexed by some real-valued parameter c. The technical complications of such an extension are beyond the scope of this paper.

It is evident that any CSI joint p.d. must be positive semidefinite, but we are so far unable to establish whether the converse holds, even in the finite-dimensional case. That is, we do not know whether all positive semidefinite finite joint probability distributions are CSI.

4 Pair Hidden Markov Models

A pair hidden Markov model (PHMM) is an HMM that generates two symbol sequences simultaneously; the two sequences need not necessarily be of the same length. The PHMM, therefore, defines a joint probability distribution over finite symbol sequences. Models of this type are used in bioinformatics to construct probabilistic models of relatedness of pairs of protein or DNA sequences, as described in [DEKM98]. A PHMM is defined as follows.

- ◇ a finite set S of states, which is the disjoint union of four subsets:

  S^{AB} — states that emit two symbols, one for A and one for B

  $\mathcal{S}^{A}$ — states that emit one symbol only for A

  $\mathcal{S}^{B}$ — states that emit one symbol only for B

  $\mathcal{S}^{-}$ — states that emit no symbols

- ◇ Distinguished states START and END. The process starts in START, and ends in the absorbing state END. For notational reasons, it will be convenient to define that START, END $\in \mathcal{S}^{AB}$, but both START and END emit no symbols.

- ◇ A function T that gives state transition probabilities: $T(\mathsf{s},\mathsf{t})$ is the probability that the next state is t given that the current state is s.

- ◇ An alphabet $\mathcal{B}$

- ◇ For states that emit symbols, probability distributions over $\mathcal{B}$:
  - For each state $\mathsf{s} \in \mathcal{S}^{AB}$, a probability distribution over $\mathcal{B} \times \mathcal{B}$
  - For each state $\mathsf{s} \in \mathcal{S}^{A}$or $\mathcal{S}^{B}$ a probability distribution over $\mathcal{B}$

The class $\mathcal{S}^{-}$ of non-emitting states is included for notational convenience: all states in $\mathcal{S}^{-}$ can be eliminated with no change to the joint distribution of emitted sequences.

A *realisation* of the PHMM is a sequence of states, starting with START and finishing with END , together with the symbol(s), if any, emitted in each state. Each realisation, therefore, is a complete record of the construction of two particular sequences a and b.

The probability of any one realisation of a PHMM is straightforward to calculate. But any particular pair of sequences a and b may be generated by exponentially many different realisations. Happily there are well-known efficient dynamic programming algorithms for summing over all possible realisations to calculate the joint probability of any two particular sequences a and b.

The point of using a PHMM is that it is easy to compute joint probabilities of pairs of sequences. Under what circumstances can this joint probability be represented as a dot-product and used as a kernel?

5 Conditionally Symmetrically Independent PHMMs

The state diagram of a useful CSI PHMM is shown in 5 below.

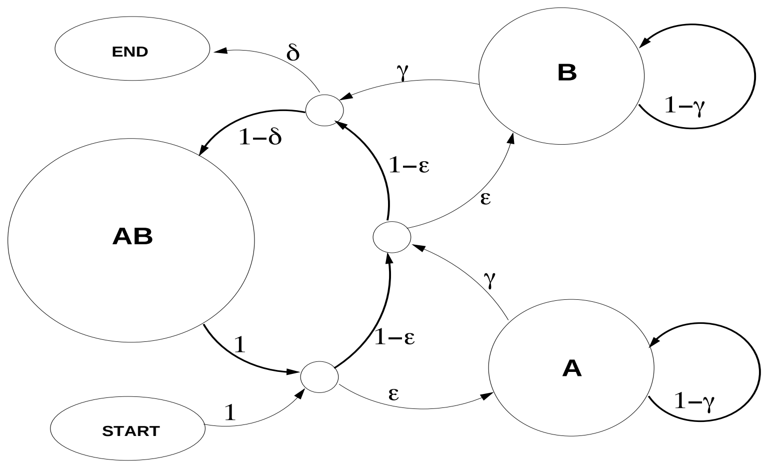

Figure 1 A CSI pair HMM for Matching

The state AB emits matching, or nearly matching symbols for both sequences; the states A and B emit *insertions*, parts of one sequence that are not parts of the other. ϵ, δ, and γ are all small probabilities. The most frequently taken state-transitions are drawn with thicker arrows. The PHMM starts in START, and then typically repeatedly cycles through AB. Occasionally it will reach the state A or B, and then generate an insertion of several symbols, before going back to AB. Eventually, the state END will be reached, and the process will stop.

This PHMM is useful even though it only has three states that emit symbols, which is the minimum number for a non-trivial PHMM. The joint distribution defined by this PHMM gives high probabilities to sequences that match along large parts of their lengths, where "match" means that pairs of corresponding symbols are generated by the state AB.

To state sufficient conditions for a PHMM $\mathcal{H}$ to be CSI requires some definitions.

Let T^{AB} be the transition probabilities restricted to $\mathcal{S}^{AB}$. That is, for $\mathsf{s}, \mathsf{t} \in \mathcal{S}^{AB}$, let $T^{AB}(\mathsf{s}, \mathsf{t})$ be the probability that, starting from s, the next state in $\mathcal{S}^{AB}$ reached is t.

Let $\mathsf{A}^{\uparrow}(\mathsf{s}, \mathsf{t})$ be the random variable denoting the possibly empty subsequence of states in $\mathcal{S}^{A}$ that the process passes through, given that the process starts in state $\mathsf{s} \in \mathcal{S}^{AB}$, and given that state t is the next state in $\mathcal{S}^{AB}$ reached. Let $\mathsf{B}^{\uparrow}(\mathsf{s}, \mathsf{t})$ be a random variable defined similarly.

Definition 3 *A PHMM $\mathcal{H}$ has the* independent insertion property *if, for all* $\mathsf{s}, \mathsf{t} \in \mathcal{S}^{AB}$, $\mathsf{A}^{\uparrow}(\mathsf{s}, \mathsf{t})$ *and* $\mathsf{B}^{\uparrow}(\mathsf{s}, \mathsf{t})$ *are independent.*

Proposition 1 *Let $\mathcal{H}$ be a PHMM such that:*

1. *The joint distribution over sequences induced by $\mathcal{H}$ is unchanged if $\mathcal{S}^{A}$ is relabelled as $\mathcal{S}^{B}$ and $\mathcal{S}^{B}$ as $\mathcal{S}^{A}$.*

2. *For all states $s \in \mathcal{S}^{AB}$, the symbol-emission joint p.d. over $\mathcal{B} \times \mathcal{B}$ is CSI.*

3. *$\mathcal{H}$ has the independent insertion property.*

Then the joint p.d. induced by $\mathcal{H}$ over pairs of sequences of symbols is CSI.

Proof

The proof is in two stages. It is shown first that any PHMM that satisfies condition 2 may be transformed into an equivalent PHMM in which all states in $\mathcal{S}^{AB}$ have symmetric independent joint emission distributions. Next, it is shown that the probability of a realisation may be factored so that sequences A and B are independent given the subsequence of states from $\mathcal{S}^{AB}$ that occurs in the realisation. The result follows.

From condition 2, it follows for each $\mathsf{s} \in \mathcal{S}^{AB}$, the symbol-emission p.d. is a mixture of symmetric independent distributions. It is possible to construct an equivalent PHMM to $\mathcal{H}$ in which all states in $\mathcal{S}^{AB}$ have symmetric independent emission distributions, by replacing each state in $\mathcal{S}^{AB}$ with a network of states.

As shown in figure 2, the state s can be decomposed into a non-emitting entry state $\mathsf{s}_{\mathsf{entry}}$, a set of alternative *atomic* doubly emitting states $\mathsf{s}_1, \mathsf{s}_2, \ldots$ and an exit state $\mathsf{s}_{\mathsf{exit}}$. The number of atomic states may be finite or countably infinite: note that even if there are infinitely many atomic states, the entire PHMM is still, by construction, equivalent to finite PHMM in the sense that it generates an identical joint p.d. over symbol sequences.

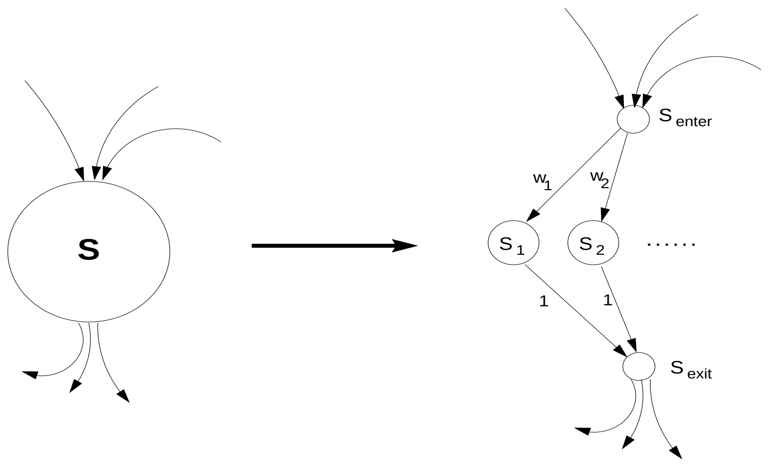

Figure 2 Re-writing a doubly emitting state as a mixture of atomic states

For each state t for which a transition to s is possible, the transition occurs to $\mathsf{s}_{\mathsf{entry}}$ with the same probability. From $\mathsf{s}_{\mathsf{entry}}$, there is a transition to one of the atomic states $\mathsf{s}_1, \mathsf{s}_2, \ldots$, the transition to s_i having probability w_i. From s_i there is a transition with probability 1 to $\mathsf{s}_{\mathsf{exit}}$, and from $\mathsf{s}_{\mathsf{exit}}$ the transition probabilities are the same as from s. The distribution of symbols emitted by the substituted network of states consisting of $\mathsf{s}_{\mathsf{entry}}$, $\mathsf{s}_1, \mathsf{s}_2, \ldots$, and $\mathsf{s}_{\mathsf{exit}}$ is exactly the same as the distribution of symbols emitted by s.

The point of this substitution is that all of the doubly emitting states $\mathsf{s}_1, \mathsf{s}_2, \ldots$ now emit pairs of *independent* symbols. From now on, therefore, we may assume that all states in $\mathcal{S}^{AB}$ emit pairs of *independent* symbols.

Let ω be a realisation of the PHMM $\mathcal{H}$. Let ω contain $n+1$ states from $\mathcal{S}^{AB}$. Let the sequence of states from $\mathcal{S}^{AB}$ be $\mathsf{c} = \langle \mathsf{c}_0, \ldots, \mathsf{c}_n \rangle$, with $\mathsf{c}_0 = \text{START}$ and $\mathsf{c}_n = \text{END}$.

Let $\mathsf{a}_i^{\uparrow}$ be the possibly empty sequence of states from $\mathcal{S}^A$ that occur between c_{i-1} and c_i in ω, and let $\mathsf{b}_i^{\uparrow}$ be defined similarly.

Let $a(\mathsf{c}_i)$ denote the symbol in sequence a emitted by the state c_i, and let $b(\mathsf{c}_i)$ be defined similarly. Let $\mathsf{a}^{\uparrow} = \left\langle \mathsf{a}_0^{\uparrow}, \ldots, \mathsf{a}_n^{\uparrow} \right\rangle$ and let $\mathsf{b}^{\uparrow} = \left\langle \mathsf{b}_0^{\uparrow}, \ldots, \mathsf{a}_n^{\uparrow} \right\rangle$ be the complete sequences of insertions of states in $\mathcal{S}^A$ and $\mathcal{S}^B$ respectively.

We seek to show that $p(a, b \mid \mathsf{c}) = p(a \mid \mathsf{c})\ p(b \mid \mathsf{c})$. Now, from the independent insertion property,

$$p(\mathsf{a}_i^{\uparrow}, \mathsf{b}_i^{\uparrow} \mid \mathsf{c}_{i-1}, \mathsf{c}_i) = p(\mathsf{a}_i^{\uparrow} \mid \mathsf{c}_{i-1}, \mathsf{c}_i)\ p(\mathsf{b}_i^{\uparrow} \mid \mathsf{c}_{i-1}, \mathsf{c}_i) \tag{15}$$

for $1 \leq i \leq n$, so that

$$\begin{aligned} p(\mathsf{a}^{\uparrow}, \mathsf{b}^{\uparrow} \mid \mathsf{c}) &= \prod_{i=1}^{n} p(\mathsf{a}_i^{\uparrow}, \mathsf{b}_i^{\uparrow} \mid \mathsf{c}_{i-1}, \mathsf{c}_i) \\ &= p(\mathsf{a}^{\uparrow} \mid \mathsf{c}) p(\mathsf{b}^{\uparrow} \mid \mathsf{c}) \end{aligned} \tag{16}$$

As each c_i is an atomic state with an independent emission distribution,

$$p(a(\mathsf{c}_i), b(\mathsf{c}_i) \mid \mathsf{c}_i) = p(a(\mathsf{c}_i) \mid \mathsf{c}_i)\ p(b(\mathsf{c}_i) \mid \mathsf{c}_i) \tag{17}$$

for $1 \leq i \leq n$, and since states in $\mathcal{S}^A$ do not affect symbols in b, and vice versa, it follows from (17) that

$$p(a, b \mid \mathsf{a}^\dagger, \mathsf{b}^\dagger, \mathsf{c}) = p(a \mid \mathsf{a}^\dagger, \mathsf{c})\ p(b \mid \mathsf{b}^\dagger, \mathsf{c}) \tag{18}$$

Hence

$$\begin{aligned} p(a, b \mid \mathsf{c}) &= \sum_{\mathsf{a}^\dagger, \mathsf{b}^\dagger} p(a, b \mid \mathsf{a}^\dagger, \mathsf{b}^\dagger, \mathsf{c})\ p(\mathsf{a}^\dagger, \mathsf{b}^\dagger \mid \mathsf{c}) && (19) \\ &= \sum_{\mathsf{a}^\dagger, \mathsf{b}^\dagger} \left(p(a \mid \mathsf{a}^\dagger, \mathsf{c}) p(\mathsf{a}^\dagger \mid \mathsf{c})\right) \left(p(b \mid \mathsf{b}^\dagger, \mathsf{c}) p(\mathsf{b}^\dagger \mid \mathsf{c})\right) && (20) \\ &= p(a \mid \mathsf{c})\ p(b \mid \mathsf{c}) && (21) \end{aligned}$$

where (20) follows from (18), (16) and rearrangement of terms. ▮

This proof shows that a natural and currently used matching function for sequences can be represented as a dot-product. The feature space has one dimension for each possible sequence of atomic doubly emitting states c; the number of such c for which the mapping $\phi(a)$ is non-zero is in general exponential in the length of the symbol sequence a.

6 Conclusion

A natural currently used class of match-scores for sequences have been shown to be representable as dot-products in a high-dimensional space. It follows that these match-scores can be used in dual formulations of linear statistical methods, and also that the match-scores may be used to locate sequences in a Euclidean space. We are investigating possible applications and extensions of this approach for bio-sequence analysis and speech recognition.

References

[DEKM98] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic models of proteins and nucleic acids.* Cambridge University Press, 1998.

[Joa97] T. Joachims. Text categorization with support vector machines. In *Proceedings of the European Conference on Machine Learning*, 1997.

[SSM98] B. Schoelkopf, A. Smola, and K.-R. Mueller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[Vap95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[Wah90] Grace Wahba. *Spline Models for Observational Data.* SIAM, CBMS-NSF Regional Conference Series, v59, 1990.