Title: String kernel

URL: https://en.wikipedia.org/wiki/String_kernel

PageID: 27618564

Categories: Category:Algorithms on strings, Category:Kernel methods for machine learning, Category:Natural language processing, Category:String metrics

---

In machine learning and data mining , a string kernel is a kernel function that operates on strings , i.e. finite sequences of symbols that need not be of the same length. String kernels can be intuitively understood as functions measuring the similarity of pairs of strings: the more similar two strings a and b are, the higher the value of a string kernel K (a , b) will be.

Using string kernels with kernelized learning algorithms such as support vector machines allow such algorithms to work with strings, without having to translate these to fixed-length, real-valued feature vectors . [1] String kernels are used in domains where sequence data are to be clustered or classified , e.g. in text mining and gene analysis . [2]

Informal introduction

Suppose one wants to compare some text passages automatically and indicate their relative similarity.

For many applications, it might be sufficient to find some keywords which match exactly.

One example where exact matching is not always enough is found in spam detection . [3] Another would be in computational gene analysis, where homologous genes have mutated , resulting in common subsequences along with deleted, inserted or replaced symbols.

Motivation

Since several well-proven data clustering, classification and information retrieval

methods (for example support vector machines) are designed to work on vectors

(i.e. data are elements of a vector space), using a string kernel allows the extension of these methods to handle sequence data.

The string kernel method is to be contrasted with earlier approaches for text classification where feature vectors only indicated

the presence or absence of a word.

Not only does it improve on these approaches, but it is an example for a whole class of kernels adapted to data structures, which

began to appear at the turn of the 21st century. A survey of such methods has been compiled by Gärtner. [4]

In bioinformatics string kernels are used especially to transform biological sequences such as proteins or DNA into vectors for further use in machine learning models. An example of a string kernel used for that purpose is the profile kernel. [5]

Definition

A kernel on a domain D {\displaystyle D} is a function K : D × D → R {\displaystyle K:D\times D\rightarrow \mathbb {R} } satisfying some conditions (being symmetric in the arguments, continuous and positive semidefinite in a certain sense).

Mercer's theorem asserts that K {\displaystyle K} can then be expressed as K (x , y) = ϕ (x) ⋅ ϕ (y) {\displaystyle K(x,y)=\varphi (x)\cdot \varphi (y)} with ϕ {\displaystyle \varphi } mapping the arguments into an inner product space .

We can now reproduce the definition of a string subsequence kernel [1] on strings over an alphabet Σ {\displaystyle \Sigma } . Coordinate-wise, the mapping is defined as follows:

The i {\displaystyle \mathbf {i} } are multiindices and u {\displaystyle u} is a string of length n {\displaystyle n} :

subsequences can occur in a non-contiguous manner, but gaps are penalized.

The multiindex i {\displaystyle \mathbf {i} } gives the positions of the characters matching u {\displaystyle u} in s {\displaystyle s} . l (i) {\displaystyle l(\mathbf {i})} is the difference between the first and last entry in i {\displaystyle \mathbf {i} } , that is: how far apart in s {\displaystyle s} the subsequence matching u {\displaystyle u} is.

The parameter λ {\displaystyle \lambda } may be set to any value between 0 {\displaystyle 0} (gaps are not allowed, as only 0 0 {\displaystyle 0^{0}} is not 0 {\displaystyle 0} but 1 {\displaystyle 1}) and 1 {\displaystyle 1} (even widely-spread "occurrences" are weighted the same as appearances as a contiguous substring, as 1 l (i) = 1 {\displaystyle 1^{l(\mathbf {i})}=1}).

For several relevant algorithms, data enters into the algorithm only in expressions involving an inner product of feature vectors,

hence the name kernel methods . A desirable consequence of this is that one does not need to explicitly calculate the transformation φ (x) {\displaystyle \phi (x)} , only the inner product via the kernel, which may be a lot quicker, especially when approximated . [1]

References