

Package ‘stringkernels’

January 2, 2012

Type Package

Title String Kernel Methods for kernlab

Version 0.8.9

Date 2010-04-11

Author Martin Kober

Maintainer Martin Kober <martin.kober@gmail.com>

Description Gapped and word-based string kernels for use with kernlab

License GPL-2

LazyLoad yes

Depends methods, kernlab, openNLP

Suggests tm (>= 0.5)

Repository CRAN

Date/Publication 2010-04-11 17:11:34

R topics documented:

| | |
|--------------------------------|---|
| gapweightkernel | 2 |
| kernelMatrix-methods | 3 |
| multigapweightkernel | 3 |
| precomputedkernel | 5 |
| stringdotEx | 7 |
| worddot | 8 |

| | |
|--------------|-----------|
| Index | 10 |
|--------------|-----------|

| | |
|-----------------|-----------------------------------|
| gapweightkernel | <i>Gap-weighted string kernel</i> |
|-----------------|-----------------------------------|

Description

Creates a kernel object for the gap-weighted string kernel. This kernel uses words as tokens by default. Use in conjunction with **kernelab**.

Usage

```
gapweightkernel(length = 2, lambda = 0.75, normalized = TRUE,
  tokenizer = openNLP::tokenize, use_characters = FALSE)
```

Arguments

| | |
|----------------|--|
| length | Match length (excluding gaps) |
| lambda | Gap length penalty factor |
| normalized | Normalize kernel values (default: TRUE) |
| tokenizer | String tokenizer function. By default, this uses openNLP 's <code>tokenize</code> to split the text into words, but users may specify their own function. Ignored if <code>use_characters</code> is TRUE. |
| use_characters | Split texts by character, rather than by word. |

Details

This kernel generation function returns a kernel that computes the number of gapped (non-contiguous) matches of length matching tokens between two strings. Gaps are penalized by a factor `lambda`, i.e., each match is assigned a weight of λ^{L-l} , with L as the total match length and l as length.

By default, this kernel uses words (and punctuation marks) rather than characters as atomic tokens. This usually yields better results than gapped character matching.

This implementation is based on the gapped substring kernel by Rousu/Shawe-Taylor. Note that this algorithm is optimized for large alphabets, usually consisting of words.

Value

An S4 object of class `stringKernelEx`.

Author(s)

Martin Kober
<martin.kober@gmail.com>

References

Juho Rousu and John Shawe-Taylor. Efficient computation of gapped substring kernels on large alphabets. *Journal of Machine Learning Research*, 6:1323-1344, 2005.

See Also

[multigapweightkernel](#)

Examples

```
s = "The cat was chased by the fat dog"
t = "The fat cat bit the dog"
gwk = gapweightkernel()
gwk(s,t)

gwk2 = gapweightkernel(length=4, normalized=FALSE)
gwk2(s,t)

gwk3 = gapweightkernel(lambda=1, normalized=FALSE)
gwk3(s,t)
```

kernelMatrix-methods *Methods for stringKernelEx objects*

Description

S4 methods for kernels of the stringkernelEx class. See [kernelMatrix](#) for details.

multigapweightkernel *Multiple gap-weight kernels*

Description

Compute gap-weight kernels of multiple length at once and pack them in a precomputed kernel.

Usage

```
multigapweightkernel(items, maxlength, kernelarray = NULL, lambda = 0.75,
  normalized = TRUE, tokenizer = openNLP::tokenize, minlength = 1)
```

```
## S4 method for signature 'multigapweight'
getkernel(mgw, length, use_dummy = FALSE)
```

Arguments

| | |
|-------------|---|
| items | List of input texts |
| maxlength | Maximum match length |
| kernelarray | Optionally supply an array of kernel values. |
| lambda | Gap length penalty factor |
| normalized | Normalize kernel values |
| tokenizer | String tokenizer function. By default, this uses openNLP 's tokenize to split the text into words, but users may specify their own function. |
| minlength | Minimum match length |
| mgw | multigapweight object returned by multigapweightkernel |
| length | The desired length parameter for the kernel |
| use_dummy | The flag use_dummy=TRUE can be used to create a kernel with dummy values (see precomputedkernel) |

Details

The dynamic programming algorithm used for the gap-weighted kernel works by computing the matching statistics for an incrementally larger match length.

Therefore, computing the kernel value for match length n does not take significantly less computational time than computing all kernel values for $n' \leq n$.

This function computes kernel matrices for multiple lengths in one step. The getkernel method retrieves the matrix of the desired length and creates a kernel object with the precomputed values.

Value

A multigapweight object that contains the kernel value array (a kernel matrix with an additional dimension for length) and the kernel parameters.

Author(s)

Martin Kober
<martin.kober@gmail.com>

See Also

[precomputedkernel](#)

Examples

```
library(tm)

## This is necessary to make tm's corpora usable with
## stringkernel's S4 classes.
setOldClass(c("VCorpus", "Corpus"))
setIs("Corpus", "list")
```

```

data(crude)

m = multigapweightkernel(crude, maxlength=3, minlength=2)

k2 = getkernel(m, 2)
k3 = getkernel(m, 3)

kernelMatrix(k2, crude[1:5])
kernelMatrix(k3, crude[1:5])

```

```
precomputedkernel      Kernel with precomputed values.
```

Description

This function creates a kernel that stores precomputed kernel values and retrieves them as needed. Use in conjunction with **kernlab**.

Usage

```

precomputedkernel(kernel, items, kernelmatrix = NULL, use_kernel = FALSE,
                  use_dummy = TRUE)

precomputeddummy(items)

```

Arguments

| | |
|--------------|--|
| kernel | May be either a string kernel object of class <code>stringkernelEx</code> or an arbitrary character string containing the name of the kernel used. In the latter case, a <code>kernelmatrix</code> has to be supplied and <code>use_kernel</code> must be <code>FALSE</code> . |
| items | Items to be compared. |
| kernelmatrix | A kernel matrix corresponding to the <code>items</code> and the <code>kernel</code> . If not provided, <code>kernel</code> is used to compute it on-the-fly. |
| use_kernel | If <code>TRUE</code> , <code>kernel</code> is called whenever the object is called with texts not found in <code>items</code> . Otherwise, only stored values are used and input values not in <code>items</code> will raise an error. |
| use_dummy | Use dummy texts instead of the actual items. When true, the kernel does not match the actual items, instead it interprets the input texts as integer indices. See Details on how to use this. Cannot be used in conjunction with <code>use_kernel</code> . |

Details

On most string kernel tasks, computing the kernel is the most time-consuming operation. This kernel can store kernel values and transparently return stored values instead of computing them on-the-fly. This is potentially useful whenever the same kernel values are needed for multiple classification runs, e.g., cross-fold validation or parameter tuning.

The kernel internally matches input items to the stored items and retrieves the stored kernel values. When corpora are S4 objects and/or very large, matching input items to stored items can consume a lot of time. The `use_dummy` flag can be used to avoid this. The flag instructs the kernel function to interpret input values directly as indices into the kernel matrix. These “dummy” input values can be created with `precomputeddummy(items)`. The dummy values are compatible with all training and prediction functions in **kernlab** that expect strings as input (see Examples).

Value

A S4 kernel object of class `stringkernelPrecomputed`.

Author(s)

Martin Kober
<martin.kober@gmail.com>

See Also

[multigapweightkernel](#)

Examples

```
library(tm)

## This is necessary to make tm's corpora usable with
## stringkernels' S4 classes.
setOldClass(c("VCorpus", "Corpus"))
setIs("Corpus", "list")

data(crude)

wdk = worddot(type="spectrum", length=2)
kernelMatrix(wdk, crude[1:3], crude[17:20])

pre = precomputedkernel(wdk, crude, use_dummy=TRUE)
dummy = precomputeddummy(crude)

kernelMatrix(pre, dummy[1:3], dummy[17:20])

class = factor(rep(c(1,-1),10))
model = ksvm(dummy[1:10], class[1:10], kernel=pre)

predict(model, dummy[11:20])
```

stringdotEx *Character-based string kernels*

Description

Character-based string kernels, analogous to **kernlab**'s `stringdot`.

Usage

```
stringdotEx(type = c("spectrum", "boundrange", "constant", "exponential"),
            length = 4, lambda = 1.1, normalized = TRUE)
```

Arguments

| | |
|------------|---|
| type | Type of kernel to be used. Four types are supported: spectrum Matches of exactly length n . boundrange Matches of all lengths up to n . exponential Matches of all lengths with exponentially decaying weighting λ^{-n} . constant Matches of all lengths with equal weighting. |
| length | Length of the substrings (only for spectrum and boundrange kernels) |
| lambda | Weighting factor, must be > 1 (only for exponential kernel) |
| normalized | Normalize word kernel values (default: TRUE) |

Details

This function is identical to the `stringdot` function in **kernlab**, but compatible with functions like `precomputedkernel` requiring a `stringkernelEx` object.

Value

An S4 kernel object of class `stringkernelEx`.

Note

All newline in the input strings are internally converted to carriage returns (`\r`).

Author(s)

Martin Kober
<martin.kober@gmail.com>

See Also

[stringdot](#)

Examples

```
s = "The cat was chased by the fat dog"
t = "The fat cat bit the dog"
stk = stringdotEx(type="spectrum", length=3, normalized=FALSE)
stk(s,t)
stk(s,s)
```

worddot

Word-based string kernels.

Description

This function is analogous to **kernlab**'s `stringdot`, using words instead of characters.

Usage

```
worddot(type = c("spectrum", "boundrange", "constant", "exponential"),
        length = 4, lambda = 1.1, normalized = TRUE,
        tokenizer = openNLP::tokenize)
```

Arguments

| | |
|------------|--|
| type | Type of kernel to be used. Four types are supported: spectrum Matches of exactly length n . boundrange Matches of all lengths up to n . exponential Matches of all lengths with exponentially decaying weighting. λ^{-n} . constant Matches of all lengths with equal weighting. |
| length | Length of the substrings (only for spectrum and boundrange kernels) |
| lambda | Weighting factor, must be > 1 (only for exponential kernel) |
| normalized | Normalize word kernel values (default: TRUE) |
| tokenizer | String tokenizer function. By default, this uses openNLP 's <code>tokenize</code> to split the text into words, but users may specify their own function. |

Details

This function is identical to the `stringdot` function in **kernlab**, only that it uses words instead of characters as tokens.

Value

An S4 kernel object of class `stringkernelEx`.

Author(s)

Martin Kober
<martin.kober@gmail.com>

See Also

[stringdot](#)

Examples

```
s = "The cat was chased by the fat dog"  
t = "The fat cat bit the dog"  
wdk = worddot(type="spectrum", length=2, normalized=FALSE)  
wdk(s,t)  
wdk(s,s)
```

Index

*Topic **methods**

kernelMatrix-methods, 3

*Topic **misc**

gapweightkernel, 2

kernelMatrix-methods, 3

multigapweightkernel, 3

precomputedkernel, 5

stringdotEx, 7

worddot, 8

gapweightkernel, 2

getkernel (multigapweightkernel), 3

getkernel, multigapweight-method
(multigapweightkernel), 3

getkernel-methods
(multigapweightkernel), 3

kernelMatrix, 3

kernelMatrix, stringkernelEx-method
(kernelMatrix-methods), 3

kernelMatrix, stringkernelPrecomputed-method
(precomputedkernel), 5

kernelMatrix-methods, 3

kernelMult, stringkernelEx, ANY-method
(kernelMatrix-methods), 3

kernelMult, stringkernelEx-method
(kernelMatrix-methods), 3

kernelMult-methods
(kernelMatrix-methods), 3

kernelPol, stringkernelEx-method
(kernelMatrix-methods), 3

kernelPol-methods
(kernelMatrix-methods), 3

multigapweight-class

(multigapweightkernel), 3

multigapweightkernel, 3, 3, 6

precomputeddummy (precomputedkernel), 5

precomputedkernel, 4, 5

show, stringkernelEx-method

(kernelMatrix-methods), 3

show-methods (kernelMatrix-methods), 3

stringdot, 7–9

stringdotEx, 7

stringkernelEx-class

(kernelMatrix-methods), 3

stringkernelPrecomputed-class

(precomputedkernel), 5

worddot, 8