

# Nonparametric Distribution Analysis for Text Mining

Alexandros Karatzoglou<sup>1</sup>, Ingo Feinerer<sup>2,3</sup>, Kurt Hornik<sup>3</sup>

<sup>1</sup>LITIS - INSA De Rouen, France

<sup>2</sup>Theory and Logic Group, Institute of Computer Languages  
Vienna University of Technology, Austria

<sup>3</sup>Department of Statistics and Mathematics  
Wirtschaftsuniversität Wien, Austria

32nd Annual GfKI Conference, 2008

# Outline

- 1 Kernel Methods
- 2 Statistical Environment
- 3 Experiments
- 4 Results
- 5 Conclusion

# String Kernels

- String Kernel function

$$k(x, x') = \sum_{s \sqsubseteq x, s' \sqsubseteq x'} \lambda_s \cdot \delta_{s, s'} = \sum_{s \in A^*} \text{num}_s(x) \cdot \text{num}_s(x') \cdot \lambda_s$$

- Number of occurrences of substrings of the same length  $n$  ( $\lambda_s = 0$  for all  $|s| \neq n$  “string kernel”)
- Number of occurrences of substrings of length up to  $n$  ( $\lambda_s = 0$  for all  $|s| > n$  “full string kernel”)

# String Kernel Types

- Constant (constant)** All common substrings are matched and weighted equally.
- Exponential decay (exponential)** All common substrings are matched but the substring weight decays as the matching substring gets shorter.
- $k$ -spectrum (spectrum)** This kernel considers only matching substrings of exactly length  $n$ , i.e.  $\lambda_s = 1$  for all  $|s| = n$ .
- Bounded range (boundrange)** A kernel where  $\lambda_s = 0$  for all  $|s| > n$  that is comparing all substrings of length less or equal to a given length  $n$ .

# Efficient Computation

- String kernels can be computed by building the suffix tree of a string  $x$  and computing the matching statistics of a string  $x'$  by traversing string  $x'$  through the suffix tree of  $x$ .
- Given a suffix tree  $S(x)$  it can be proven that the occurrence of a certain substring  $y$  can be calculated by the number of nodes at the end of the path of  $y$  in the suffix tree.
- Two main suffix tree operations are required to compute string kernels, a top down traversal for annotation and a suffix link traversal for computing matching statistics, both operations can be performed more efficiently on a suffix array.

# Suffix Arrays

- The enhanced suffix array (Abouelhoda et al., 2004) of a string  $x$  is an array of integers corresponding to the lexicographically sorted suffixes of  $x$  with additional information stored to allow for the reproduction of almost all operations available on a suffix tree.
- Suffix arrays bring the advantage of better memory use and locality thus most operations can be performed faster than on the original suffix trees.

# Kernelized Tests

- Kernel based tests for comparing distributions rely on projecting the data into Reproducing Kernel Hilbert space.
- Gretton et al. (2005) proposed a novel test, by finding a smooth function which is large on the points drawn from distribution  $p$ , and small (as negative as possible) on the points from  $q$ .
- The difference between the mean function values on the two samples is used as the test statistic.
- *Maximum Mean Discrepancy*

# Maximum Mean Discrepancy

The Maximum Mean Discrepancy (MMD) of two distributions  $p$  and  $q$  is given by

$$MMD[\mathcal{F}, p, q] = \sup_{f \in \mathcal{F}} (\mathbf{E}_{x \sim p}[f(x)] - \mathbf{E}_{y \sim q}[f(y)])$$

which can be calculated in Kernel Hilbert Space (Gretton et al., 2007) as

$$MMD^2[\mathcal{F}, p, q] = \mathbf{E}_{x, x' \sim p}[k(x, x')] - 2\mathbf{E}_{x \sim p, y \sim q}[k(x, y)] + \mathbf{E}_{y, y' \sim q}[k(y, y')]$$

where the kernel  $k$  implicitly defines the function class  $\mathcal{F}$ .

# kernlab R Package

- Extensible S4 package for kernel-based machine learning methods in R
- Contains
  - functions for computing kernel expressions
  - kernel methods for regression and classification (SVM, Gaussian Processes)
  - clustering (kernel  $k$ -means, spectral clustering)
  - ranking and kernel PCA

# tm R Package

- Text mining framework in R
- Tailored for
  - Plain texts, articles and papers
  - Web documents (XML, SGML, ...)
  - Surveys
- Methods for
  - Clustering
  - Classification
  - Visualisation

# Data

- Reuters-21578 data set (Lewis, 1997) where we extracted news articles on topics dealing with acquisitions (acq) and crude oil (crude).
- SpamAssassin public mail corpus with known classifications into normal mail (ham) and unsolicited junk mail (spam).
- Wizard of Oz books but they are very different by construction from the other two data sets.

# Experiment Setup

We used a wide variety of string kernels

- constant kernel
- spectrum kernel
- exponential kernel
- bounded range kernel

and parameter settings (e.g., lengths of 2, 4, 6, and 8) but also

- linear (vanilla) kernel

with bag-of-words term-document matrices for comparison.

# Results

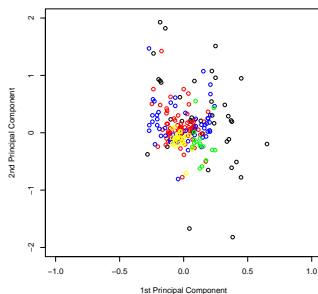
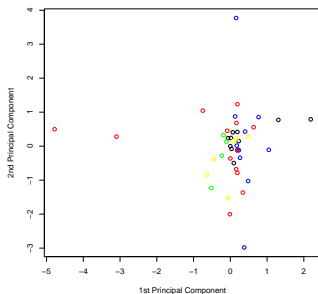
Data set	Type	Arg	H0	AsymH0	Bound
CrudeVsCrude	spectrum	4	FALSE	FALSE	0.048
CrudeVsCrude	spectrum	6	NA	NA	0.037
CrudeVsCrude	spectrum	8	FALSE	FALSE	0.053
CrudeVsCrude	spectrum	10	NA	NA	0.053
AcqVsSpam	linear	tf	FALSE	FALSE	315.530
AcqVsSpam	linear	bin	FALSE	TRUE	1.998

- Validation with distribution tests between same data sets.
- Bag-of-words does not always work reliable in our tests.

# Results

Data set	Type	Arg	H0	AsymH0	Bound
CrudeVsAcq	exponential	6	FALSE	TRUE	0.011
CrudeVsAcq	exponential	8	FALSE	TRUE	0.009
CrudeVsAcq	exponential	10	FALSE	TRUE	0.009
CrudeVsAcq	constant	6	FALSE	TRUE	0.041
CrudeVsAcq	constant	8	FALSE	TRUE	0.037
CrudeVsAcq	constant	10	FALSE	TRUE	0.041
CrudeVsAcq	spectrum	6	FALSE	TRUE	0.041
CrudeVsAcq	spectrum	8	FALSE	TRUE	0.038
CrudeVsAcq	spectrum	10	FALSE	TRUE	0.042
CrudeVsAcq	boundrange	6	FALSE	TRUE	0.008
CrudeVsAcq	boundrange	8	FALSE	TRUE	0.008
CrudeVsAcq	boundrange	10	FALSE	TRUE	0.010

# Principal Component Analysis



- Wizard of Oz books differ in various ways from other data.
- We were not able to clearly distinguish the two authors of the Oz series (Baum and Thompson).
- Principal Component Analysis gives some visual feedback.

# Running Time

- Kernel matrix computation takes longest (several hours for big data sets)
- MMD is rather fast (from seconds to minutes)
- Term-document matrix construction is fast (but results do not persuade)
- String kernels with fixed ranges and decay factors are (obviously) faster
- Spectrum kernel yields best results comparing distribution test results and running time

# Conclusion

- MMD with string kernels works quite well
- We now have a fast implementation in R (together with a text mining infrastructure and kernel lab)
- Technique is highly useful for authorship attribution, stylometry, and linguistic forensics
- However, we want to improve both the technique as its implementation, and conduct further tests
- Thanks for your attention