TECHNISCHE UNIVERSITÄT DARMSTADT

Hash Kernels

Jan Köhler

Data Mining & Machine Learning Seminar 2016

Professor Fürnkranz, Eneldo Loza Mencía

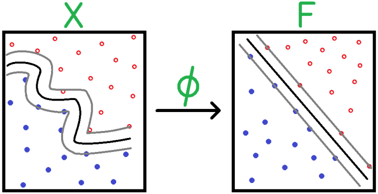
Table of contents

TECHNISCHE UNIVERSITÄT DARMSTADT

Table of contents

TECHNISCHE UNIVERSITÄT DARMSTADT

Introduction: Kernel methods
Overview



- Data is not linear classifiable in **domain of observations** X

$\rightarrow$ Data is transformed in high-dimensional **feature space** F by non-linear **feature map** $\phi : X \rightarrow F$

- Linear classifier can be applied implicitly in F

$\rightarrow$ **Kernel methods** generalize linear algorithms

- Calculations in high-dimensional feature space F are difficult

$\rightarrow$ Efficient calculation of inner products with **kernel function** $k : X \times X \rightarrow \mathbb{R}$ satisfying **kernel-trick** $k(x, x') = \langle \phi(x), \phi(x') \rangle$

Introduction: Kernel methods
Details

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kernel function $k : X \times X \to \mathbb{R}$ measures similarity between observations x and x'
- Kernel function k is generalisation of positive definite function or matrix
- Allows to operate in high-dimensional feature space F implicitly by computing the inner product of images of data

$$k(x, x') = \langle \phi(x), \phi(x') \rangle \qquad \text{(kernel-trick,}(\star))$$

  instead of its coordinates in F
- $\forall$ kernel function k $\exists$ feature space F and feature map $\phi : X \to F$ with $(\star)$

$\rightarrow$ Every algorithm that only uses inner products can be used for kernel methods

Introduction: Kernel methods
Examples

Different kernels ...

- Polynomial kernel

$$k(x, x') := \langle x, x' \rangle^d$$

- Radial basis function (RBF) kernel

$$k(x, x') := \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

- ...

... can be applied to different algorithms

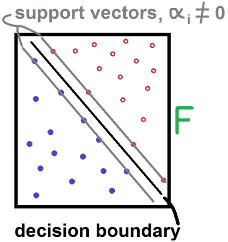
- Principal component analysis (PCA)
- Support vector machine (SVM):

  $f(x) = \text{sgn}\left(\langle w, x \rangle + b\right)$

  $f(x) = \text{sgn}\left(\langle w, \phi(x) \rangle + b\right)$ Transformation

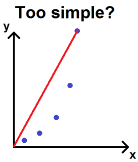
  $f(x) = \text{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i k(x_i, x) + b\right)$ Kernel-trick

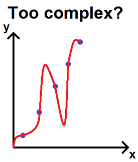
- ...



Introduction: Kernel methods
Advantages

TECHNISCHE UNIVERSITÄT DARMSTADT





- Kernel methods use rich class of functions because of feature map ϕ
- Complexity is reduced because of mathematical equivalence to linear algorithm

- Every algorithm that only uses inner products can be used (often fulfilled)
- Achieve very good results in different machine learning problems (character recognition, text categorisation, ...)

Table of contents

TECHNISCHE UNIVERSITÄT DARMSTADT

Introduction: Hash Kernels

TECHNISCHE UNIVERSITÄT DARMSTADT

- Problem: Domain of observations X is already large / high-dimensional

$\rightarrow$ Not enough memory,
especially for transformation in higher-dimensional feature space F

- Solution: **hashing-trick**

$\rightarrow$ Hashing high-dimensional data X into lower dimensional feature space F by feature map $\phi : X \rightarrow F$

$\rightarrow$ Solves memory problems

$\rightarrow$ Preserves sparsity

$\rightarrow$ No loss of information?

Introduction: Hash Kernels

Keeping the kernel simple:

- Expansion (transformation) into feature space $\phi : X \rightarrow F$ ...

$$k(x, x') = \langle \phi(x), \phi(x') \rangle$$

- ... is approximated by mapping $\overline{\phi} : X \rightarrow \overline{F}$

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle$$

- $\overline{\phi}$ has better computational properties than ϕ (e.g. sparse)

Previous Work

- Generic Randomization with sampling
  (Kontorovich, 2007; Rahimi, Recht, 2008)
- Hashing methods:
  - Randomized projections (Indyk, Motwani, 1998)
  - Count-Min Sketch (Cormode, Muthukrishnan, 2004)
  - Vowpal Wabbit learning software (Langford, 2007)
  - Random Feature Mixing (Ganchev, Dredze, 2008)

Table of contents

TECHNISCHE UNIVERSITÄT DARMSTADT

New method: Hash Kernels
Kernel Approximation

TECHNISCHE UNIVERSITÄT DARMSTADT

- $\mathcal{J}$ index set
- $h : \mathcal{J} \to \{1, ..., n\}$
  hash function from a distribution of pairwise independent hash functions
- $\phi : X \to \mathbb{R}^{|\mathcal{J}|}, \quad \phi_i(x),\ i = 1, ..., |\mathcal{J}|$ can be computed efficiently
- Approximated hash kernel:

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \quad \text{with} \quad \overline{\phi}_j(x) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=j}} \phi_i(x), \ \ j \in \{1, ..., n\}$$

$\rightarrow$ Coordinate $\overline{\phi}_j(x)$ are accumulated coordinates $\phi_i(x)$ for which $h(i) = j$

- Claim: preserves information with less computation

New method: Hash Kernels
Example: Strings

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \quad \text{with} \quad \overline{\phi}_j(x) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=j}} \phi_i(x), \quad j \in \{1, ..., n\}$$

- $X = \mathcal{J}$ domain of strings
- $\phi_i(x) := \lambda_i \#_i(x)$, coefficient $\lambda_i \geq 0$, $\#_i(x)$ number of occurences of substring i
- Kernel

$$k(x, x') = \langle \phi(x), \phi(x') \rangle = \sum_{i \in \mathcal{J}} \lambda_i^2 \#_i(x) \#_i(x')$$

New method: Hash Kernels
Example: Strings

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \quad \text{with} \quad \overline{\phi}_j(x) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=j}} \phi_i(x), \quad j \in \{1, ..., n\}$$

- $X = \mathcal{J} = \{A, B, C, AA, AB, ..., CC, AAA, ..., CCC\}$
- $\phi_i(x) := \lambda_i \#_i(x) = \#_i(x)$
- $h : \mathcal{J} \to \{1, 2, 3\}$ hashes string to its length

$$\overline{\phi}_1(AB) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=1}} \#_i(AB) = \#_A(AB) + \#_B(AB) + \#_C(AB)$$

New method: Hash Kernels
Example: Strings

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \quad \text{with} \quad \overline{\phi}_j(x) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=j}} \phi_i(x), \quad j \in \{1, ..., n\}$$

- $X = \mathcal{J} = \{A, B, C, AA, AB, ..., CC, AAA, ..., CCC\}$
- $\phi_i(x) := \lambda_i \#_i(x) = \#_i(x)$
- $h : \mathcal{J} \to \{1, 2, 3\}$ hashes string to its length

$$\overline{\phi}_1(AB) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=1}} \#_i(AB) = \#_A(AB) + \#_B(AB) + \#_C(AB) = 1 + 1 + 0 = 2$$

New method: Hash Kernels
Example: Strings

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\overline{k}(x, x') = \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \quad \text{with} \quad \overline{\phi}_j(x) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=j}} \phi_i(x), \;\; j \in \{1, ..., n\}$$

- $X = \mathcal{J} = \{A, B, C, AA, AB, ..., CC, AAA, ..., CCC\}$
- $\phi_i(x) := \lambda_i \#_i(x) = \#_i(x)$
- $h : \mathcal{J} \rightarrow \{1, 2, 3\}$ hashes string to its length

$$\overline{\phi}_1(AB) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=1}} \#_i(AB) = \#_A(AB) + \#_B(AB) + \#_C(AB) = 1 + 1 + 0 = 2$$

$$\overline{\phi}_2(AB) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=2}} \#_i(AB) = \#_{AA}(AB) + ... + \#_{CC}(AB) = 0 + 1 + 0 + ... + 0 = 1$$

$$\overline{\phi}_3(AB) = \sum_{\substack{i \in \mathcal{J} \\ h(i)=3}} \#_i(AB) = \#_{AAA}(AB) + ... + \#_{CCC}(AB) = 0 + ... + 0 = 0$$

New method: Hash Kernels
Example: Strings

TECHNISCHE UNIVERSITÄT DARMSTADT

- Computing kernel $k(x, x')$ needs $O(|x| + |x'|)$ for each pair x, x'
- Requires large amounts of working memory, especially for millions of documents

$\rightarrow$ Hashing reduces dimensionality from $|\mathcal{J}|$ to n

$\rightarrow$ Most $\phi_i(x)$ are zero, $\overline{\phi}_j(x)$ have increased density

$\rightarrow$ $\overline{\phi}(x)$ can be computed in preprocessing and x be discarded

$\rightarrow$ Memory efficient computation of kernel

Other examples

- Multiclass
- Data streams (e.g. with graphs)

Table of contents

Analysis: Bias

TECHNISCHE UNIVERSITÄT DARMSTADT

Bias of approximation $\overline{\phi}(x)$ of $\phi(x)$:

$$\begin{aligned}\overline{k}^h(x,x') &= \left\langle \overline{\phi}(x), \overline{\phi}(x') \right\rangle \\ &= \sum_j \sum_{i:h(i)=j} \phi_i(x) \sum_{i':h(i')=j} \phi_{i'}(x') \\ &= k(x,x') + \sum_{i,i':i\neq i'} \phi_i(x)\phi_{i'}(x')\delta_{h(i),h(i')}\end{aligned}$$

$$\mathbf{E}_h\left[\overline{k}^h(x,x')\right] = (1-\frac{1}{n})k(x,x') + \frac{1}{n}\sum_i \phi_i(x)\sum_{i'}\phi_{i'}(x')$$

$\rightarrow$ $\overline{k}(x,x')$ is biased estimator of kernel matrix

$\rightarrow$ Bias decreases $O(\frac{1}{n})$

Analysis: Variance

Variance of hash kernel:

$$\text{Var}\left[\overline{k}^h(x, x')\right] = \frac{n-1}{n^2}\left(k(x,x)k(x',x') + k^2(x,x') - 2\sum_i \phi_i^2(x)\phi_i^2(x')\right)$$

$\rightarrow$ Variance decreases $O(\frac{1}{n})$

$\rightarrow$ $O(\frac{1}{\sqrt{n}})$ convergence to expected value of kernel

Analysis: Information loss

TECHNISCHE UNIVERSITÄT DARMSTADT

- Hashing causes loss of information

→ Prevented with *c* duplicates of each feature

→ Information loss only if all duplicates collide with another feature

Theorem

For a random feature mapping, *l* features duplicated *c* times into a space of size *n*, the probability that all features have at least one duplicate colliding with no other features is at least

$$p \geq 1 - l\left[1 - \left(1 - \frac{c}{n}\right)^c + \left(\frac{lc}{n}\right)^c\right]$$

[3], page 499

Example:
$l = 10^5$ features,
$n = 10^8$ space size

c	p	Comment
2	59,6%	Too less duplicates
3	98,8%	Best value
10	90,0%	Too many duplicates

Analysis: Rate of convergence

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem

If

$$P\left[\left|\overline{k}^h(x,x') - \mathbf{E}_h\left[\overline{k}^h(x,x')\right]\right| > \epsilon\right] \leq c\exp\left(-c'\epsilon^2 n\right)$$

then the error for m observations and M classes is bounded by

$$\epsilon \leq \sqrt{(2\log(m+1) + 2\log(M+1) - \log(\delta) - c'')/c'}$$

[3], page 500

Table of contents

Experiments: Reuters Articles Categorisation

TECHNISCHE UNIVERSITÄT DARMSTADT

Dataset	#Train	#Test	#Label
RCV1	781.265	23.149	2

- Features: term frequency / inverse document frequency (TF-IDF)

→ Large feature dimensionality

→ Large dictionary needs to be maintained

- Solution: Hash Kernels with stochastic gradient descent (SGD)

→ Words produce hash keys: index for TF vector

→ Vocabulary can be discarded

→ IDF approximated with smaller part of training set

Experiments: Reuters Articles Categorisation

Comparison of Hash Kernels (HK) with ...

- ... Leon Bottou's Stochastic Gradient Descent SVM (BSGD)
- ... Vowpal Wabbit (VW)
- ... Vowpal Wabbit using cache file (VWC)

Algorithm	Pre	TrainTest	Error %
BSGD	303,60s	10,38s	6,02
VW	303,60s	510,35s	5,39
VWC	303,60s	5,15s	5,39
HK	0s	25,16s	5,60

[3], page 501

$\rightarrow$ No Preprocessing: Hash Kernels generates features online

$\rightarrow$ Fast performance with low error rate

Experiments: Reuters Articles Categorisation

TECHNISCHE UNIVERSITÄT DARMSTADT

Influence of hash size *n*

bits	#unique	Collision %	Error %
24	285.614	0,82	5,586
22	278.238	3,38	5,655
20	251.910	12,52	5,594
18	174.776	39,31	5,655
16	64.758	77,51	5,763
14	16.383	94,31	6,096

[3], page 501

→ Smaller hash size increases collision and error rate

→ 18 bit hash (39% collision) has similar error rate as 24 bit hash (1% collision)

→ Saves memory capacity

Experiments: Dmoz Websites Multiclass Classification

TECHNISCHE UNIVERSITÄT DARMSTADT

Dataset	#Train	#Test	#Label
Dmoz L2	4.466.703	138.146	575
Dmoz L3	4.460.273	137.924	7.100

- Topic categorization of websites using ontology DMOZ (level 2 and 3)
- Storage $O(MI)$ for M classes and I features

- Solution: Hash Kernels

$\rightarrow$ Hashing features

$\rightarrow$ Hashing features and labels jointly

Experiments: Dmoz Websites Multiclass Classification

TECHNISCHE UNIVERSITÄT DARMSTADT

Comparison of ...

- ... Hash Kernels with hashing features and labels jointly (HLF)
- ... Hash Kernels with hashing features only (HF)
- ... Baseline methods:
  - Direct model (no hash)
  - Uniform classifier (U base)
  - Majority vote (P base)

	HLF 28 bit		**HLF 24 bit**		**HF**		**no hash**	**U base**	**P base**
	error	memory	error	memory	error	memory	memory	error	error
L2	30,12	2G	30,71	0,125G	31,28	2,25G (19bit)	7,85G	99,83	85,05
L3	52,1	2G	53,36	0,125G	51,47	1,73G (15bit)	96,95G	99,99	86,83

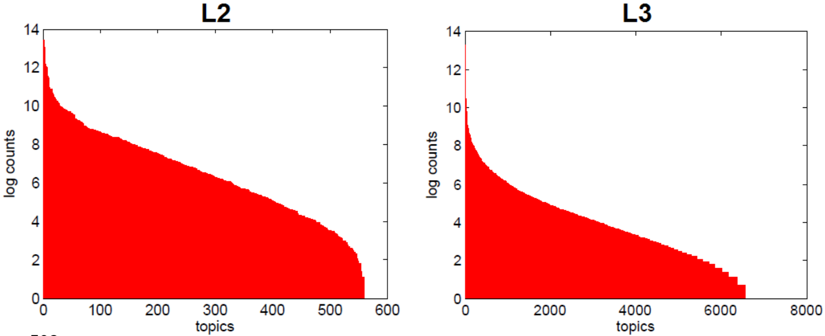
[3], page 502

$\rightarrow$ Joint hashing of features and labels is best approach

Experiments:
Dmoz Websites Multiclass Classification

TECHNISCHE UNIVERSITÄT DARMSTADT

- Frequency counts for topics on training set



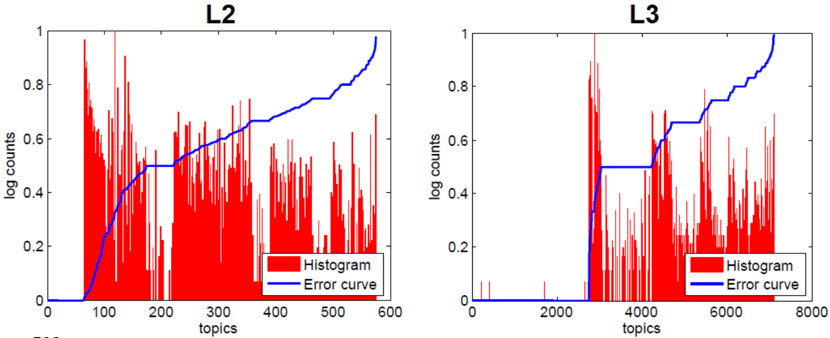
[3], page 502

→ Exponential decay in counts

Experiments:
Dmoz Websites Multiclass Classification

TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Frequency counts for topics and error on test set



[3], page 502

→ Error evenly distributed among size of classes

→ Near empty classes are learned perfectly

Table of contents

Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

Hash Kernels ...

- ... reduce computational effort by transformation into lower dimensional feature space F
- ... avoid loss of information with duplication
- ... make possible multiclass classification with large amount of classes and features
- ... have good theoretical and practical results

- Bernhard Schölkopf; Klaus-Robert Müller; Alexander J. Smola.
  Lernen mit Kernen.
  Springer-Verlag: Informatik - Forschung und Entwicklung, 1999.
- Kilian Weinberger; Anirban Dasgupta; Josh Attenberg; John Langford; Alex Smola.
  Feature hashing for large scale multitask learning.
  2821 Mission College Blvd., Santa Clara, CA 95051 USA, 2010. International Conference on Machine Learning (ICML).
- Q. Shi; J. Petterson; G. Dror; J. Langford; A. Smola; A. Strehl; V. Vishwanathan.
  Hash kernels.

volume 5, Clearwater Beach, Florida, USA, 2009. 12th International Conference on Artificial Intelligence and Statistics (AISTATS), Journal of Machine Learning: W & CP 5.

Thanks for your attention!