
Text Categorization

Luboš Popelínský

Faculty of Informatics, Masaryk University Brno

popel@fi.muni.cz, www.fi.muni.cz/~popel

Based partially on

Raymond J. Mooney's course of Machine learning

<http://www.cs.utexas.edu/~mooney/#Teaching>

University of Texas at Austin, and

Jose Maria Gomes Hidalgo's ECML/PKDD tutorial

www.fi.muni.cz/~popel/nll, and maybe

Tom Mitchell's handouts to ML book

Text Categorization and Text Mining

- Text mining
- Unsupervised learning
 - words:
e.g. Latent Semantic Analysis, Key Phrase Extraction
 - documents:
e.g. Document clustering, Topic Detection
- Supervised learning
 - words:
e.g. Part-of-Speech Tagging, Word Sense Disambiguation
 - documents:
e.g. Text categorization/classification/filtering
Information Extraction

Word Sense Disambiguation

Fui a um *casamento*.

Casamento é uma experiência interessante.

Ela me *contou* tudo.

Eu *contei* pelo menos vinte pessoas.

Você pode *contar* comigo.

Isto não está *direito*.

Ela está na faculdade de *direito*.

... *fazer* ...

Other Applications

Context-sensitive spelling checking – peace of chocolate

Named entity recognition – name, date, location, \$, @, URL

Coreference resolution – Carlos não fica na casa. El ...

Sentiment classification – positive or negative

Text summarisation

Example

spam
legit
spam spam
legit legit
spam spam
legit

Category

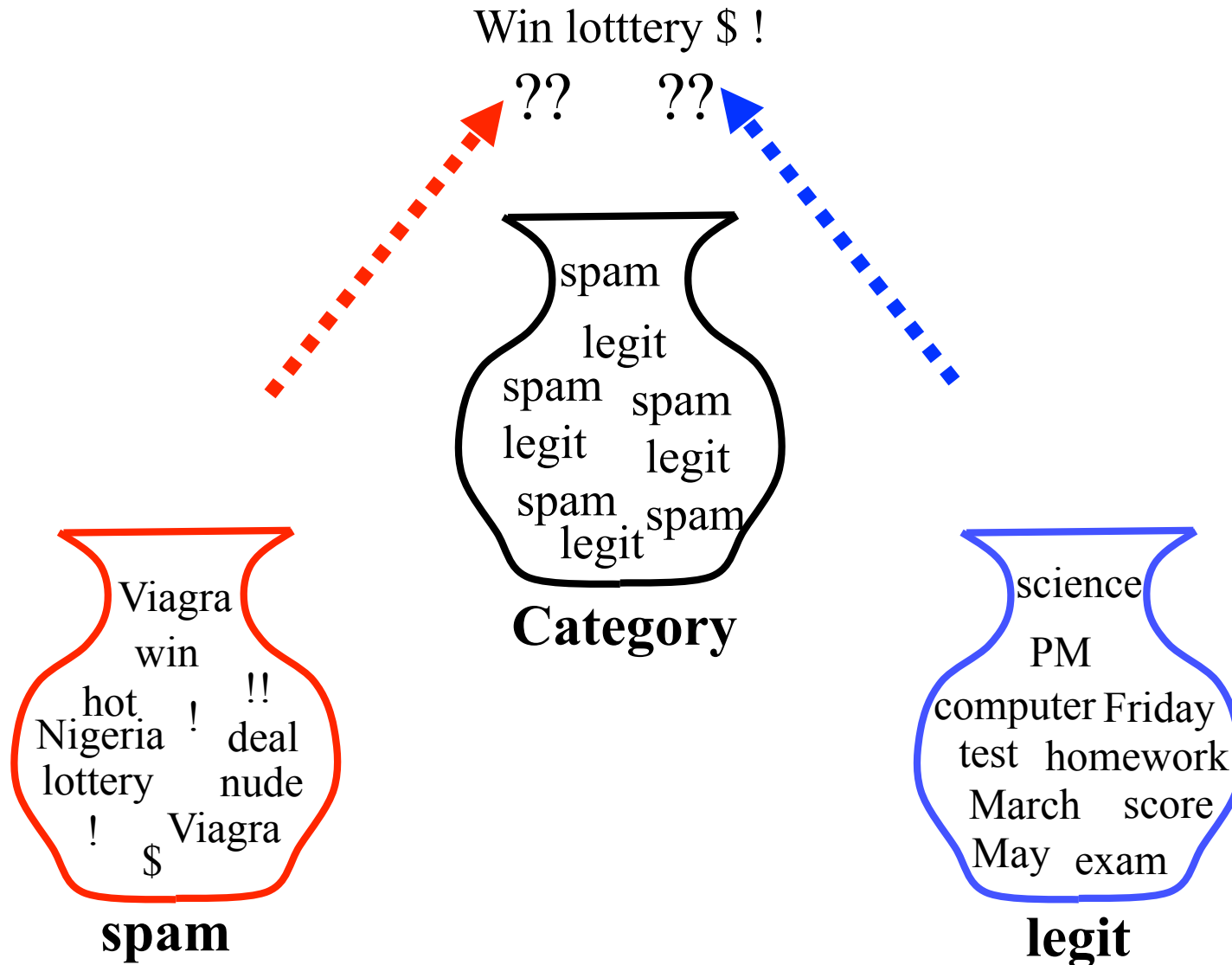
Viagra
win
hot !!
Nigeria ! deal
lottery nude
! Viagra
\$

spam

science
PM
computer Friday
test homework
March score
May exam

legit

Example (cont.)



Text Categorization Applications

- Web pages
 - Recommending
 - Yahoo-like classification
- Newsgroups/Emails
 - Recommending
 - spam filtering
 - Sentiment analysis for marketing
- News articles
 - Personalized newspaper
- Scientific papers
 - Filtering

Task

Based on the text of a **document** in a **collection**,
assign to each document **one or more labels**

Usable e.g. for

topic recognition

genre detection

authorship recognition

author's meaning (agree/disagree, against/for)

language recognition

Benchmarks: **20 Newsgroup, Reuters data**

Text Categorization

- The dominant approach is

Given a set of manually classified (labeled) documents

Use ML techniques to induce an automatic classifier of new documents

- See [Sebastiani 02] for an in-depth survey

Bag of words

Documents represented as **bag of words**

a **structure** of documents, including sentences, **is ignored**

Decision of a classifier =

based on word frequency in positive and negative examples

What words to choose to reach the best accuracy?

Maybe all ...

Text Categorization Methods

- Representations of text are very **high dimensional** (one feature for each word).
- Vectors are **sparse** since most words are rare.
- For most text categorization tasks, there are **many irrelevant** and many relevant **features**.
- Methods that sum evidence from many or all features (e.g. **naïve Bayes, neural-net, SVM**) tend to work better than ones that try to isolate just a few relevant features (decision-tree or rule induction).

Text Categorization Algorithm

- Select terms (words, bi-, trigrams, named entities)
- Perform morphological analysis (lemmatization, tagging)
lemmatization = sou, es → ser
- Term clustering = ser, estar, ficar → artificial term SEF
- Remove nonsignificant terms (stoplist)
- Select terms with highest discriminative power
- Learn classifier

Naïve Bayes for Text

- Modeled as generating a bag of words for a document in a given category by repeatedly sampling with replacement from a vocabulary $V = \{w_1, w_2, \dots, w_m\}$ based on the probabilities $P(w_j | c_i)$.
- Smooth probability estimates with **Laplace m -estimates** assuming a uniform distribution over all words ($p = 1/|V|$) and $m = |V|$
 - Equivalent to a virtual sample of seeing each word in each category exactly once.

Text Naïve Bayes Algorithm (Train)

Let V be the vocabulary of all words in the documents in D

For each category $c_i \in C$

Let D_i be the subset of documents in D in category c_i

$$P(c_i) = |D_i| / |D|$$

Let T_i be the concatenation of all the documents in D_i

Let n_i be the total number of word occurrences in T_i

For each word $w_j \in V$

Let n_{ij} be the number of occurrences of w_j in T_i

$$\text{Let } P(w_j | c_i) = (n_{ij} + 1) / (n_i + |V|)$$

Text Naïve Bayes Algorithm (Test)

Given a test document X

Let n be the number of word occurrences in X

$$\operatorname{argmax}_{c_i \in C} P(c_i) \prod_{i=1}^n P(a_i | c_i)$$

Return the category:

where a_i is the word occurring the i th position in X

Text Document Filtering

Example: filtering interesting news from a news group

Idea:

1. label interesting documents (positive examples)
2. and noninteresting ones (negative examples)
3. run a learning algorithm

also useful for spam detection and filtering web pages

Applications

Spam filtering

Sentiment analysis

Recommender systems

e.g. to recommend conference papers, or web pages

Another example: Prediction of stock price movement from news

...

Text Categorization: Summary

- Bag of words -> Vector model
- Feature/Term selection and Term extraction
- Use ML algorithm, maybe not only Naïve Bayes Classifier, SVM or neural nets

Textual Similarity Metrics

- Measuring similarity of two texts is a well-studied problem.
- Standard metrics are based on a “bag of words” model of a document that ignores word order and syntactic structure.
- May involve removing common “stop words” and stemming to reduce words to their root form.
- Vector-space model from Information Retrieval (IR) is the standard approach.
- Other metrics (e.g. edit-distance) are also used.

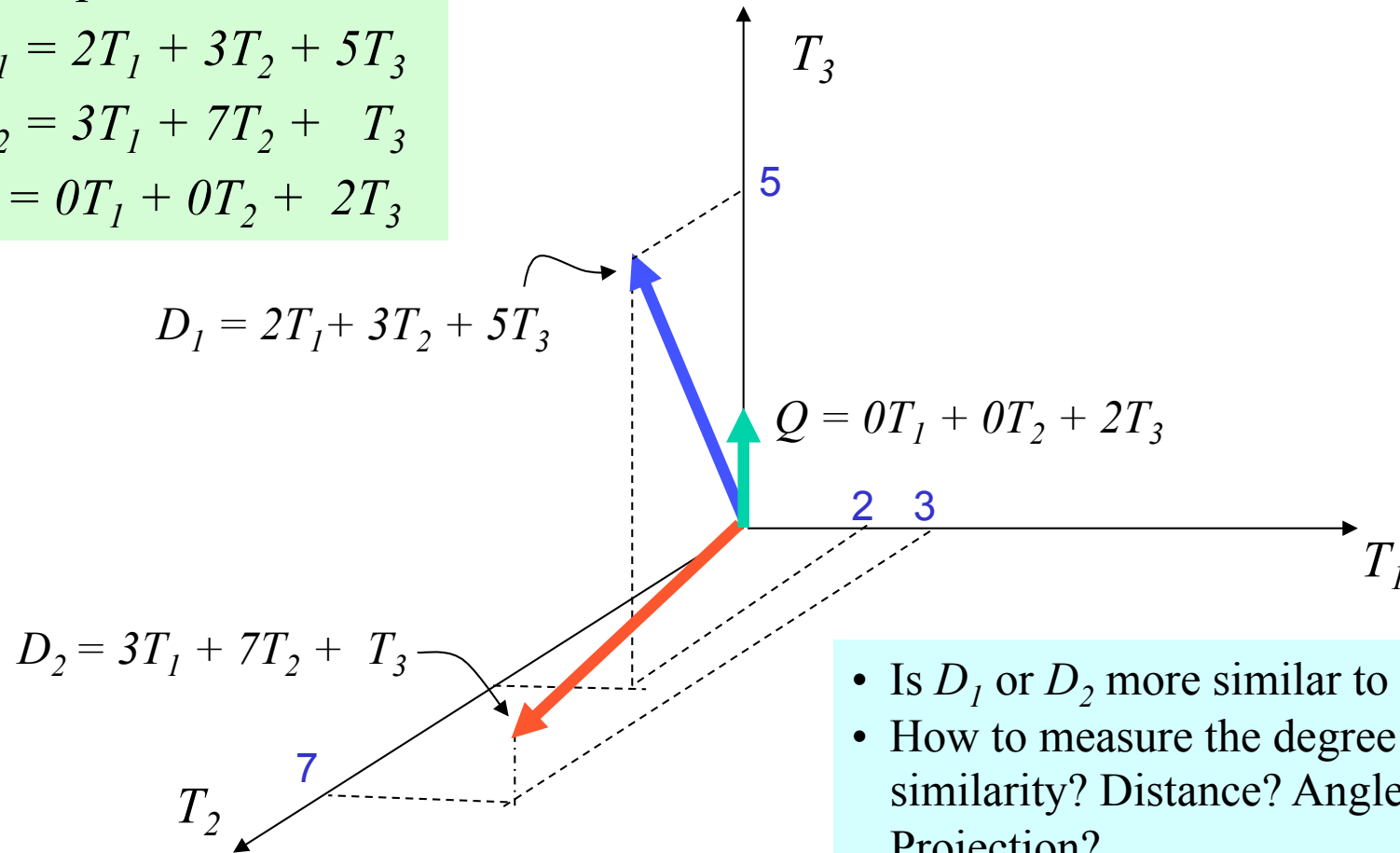
Graphic Representation

Example:

$$D_1 = 2T_1 + 3T_2 + 5T_3$$

$$D_2 = 3T_1 + 7T_2 + T_3$$

$$Q = 0T_1 + 0T_2 + 2T_3$$



- Is D_1 or D_2 more similar to Q ?
- How to measure the degree of similarity? Distance? Angle? Projection?

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

$$f_{ij} = \text{frequency of term } i \text{ in document } j$$

- May want to normalize *term frequency* (*tf*) by dividing by the frequency of the most common term in the document:

$$tf_{ij} = f_{ij} / \max_i \{f_{ij}\}$$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

df_i = document frequency of term i

= number of documents containing term i

idf_i = inverse document frequency of term i ,

= $\log_2 (N / df_i)$

(N: total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to tf .

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

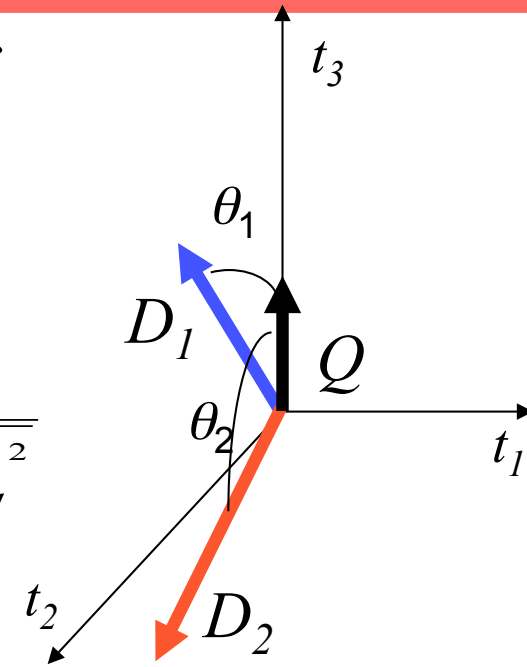
$$w_{ij} = tf_{ij} idf_i = tf_{ij} \log_2 (N/ df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

Cosine Similarity Measure

- Cosine similarity measures the cosine of the angle between two vectors.
- Inner product normalized by the vector lengths.

$$\text{CosSim}(d_j, q) = \frac{\mathbf{r}_{d_j} \cdot \mathbf{r}_q}{|\mathbf{r}_{d_j}| \cdot |\mathbf{r}_q|} = \frac{\sum_{i=1}^t (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^t w_{ij}^2 \cdot \sum_{i=1}^t w_{iq}^2}}$$



$$D_1 = 2T_1 + 3T_2 + 5T_3 \quad \text{CosSim}(D_1, Q) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$$

$$D_2 = 3T_1 + 7T_2 + 1T_3 \quad \text{CosSim}(D_2, Q) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$$

$$Q = 0T_1 + 0T_2 + 2T_3$$

D_1 is 6 times better than D_2 using cosine similarity but only 5 times better using inner product.