Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

http://www.esi.uem.es/~jmgomez/tutorials/ecmlpkdd02/
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

0

OUTLINE
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

Goals and methodology

• An overview of Text Mining ...
• ... exploring two Internet content filtering applications ...
• ... following the standard KDD process and ...
• ... working with operational code
Outline I

1. Text Mining: What is it and what is it not?
2. Learning from text when we know what about to learn
3. Learning from text when we do not know what about to learn
4. Tools for Text Mining

Outline II

5. Application to the detection of offensive websites
6. Application to the detection of unsolicited bulk email
7. Challenges in Text Mining
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

1

TEXT MINING
Introduction I

- Attractive field due to the Internet / Intranet / Digital Libraries explosion
- Increasing amount of text in electronic form
- E.g. [Moore 00], by July
  - Number of unique pages on Internet: 2.1 billion
  - Unique pages added per day: 7.3 million
- E.g. [Oracle 97]
  - text represents about 90% of all information handled by an organization
Introduction II

• About the name(s)
  – Text Mining, Text Data Mining, Knowledge Discovery in Text, Knowledge Discovery in Textual Data(bases)
  – Like Data Mining as a step in the KDD process [Fayyad et al. 96], we could see the Text Mining step in the Knowledge Discovery in Textual Data process
  – We will take the sense by Hearst [Hearst 99]

Definition of Text Mining I

• Several definitions in the literature (e.g. [Dörre et al. 99, Feldman & Dagan 95, Hearst 99, Kodratoff 99, Rajman & Besaçon 98])
• Extending the KDD definition, Text Mining is “the nontrivial extraction of implicit, previously unknown, and potentially useful information from (large amounts of) textual data”
Definition of Text Mining II

- To what extent is something *previously unknown*?
  - In Hearst’s opinion, nor even the writer knows => real new knowledge = real Text Mining
    - E.g. The discovery of an *absolutely new*, potentially effective treatment for a disease by exploring scientific literature
    - E.g. The discovery of the fact that private and not public funding leads to more inventions by exploring patent files
  - Others think we have to rediscover the information the author encoded in text

Definition of Text Mining III

- But to get knowledge, users must be helped to locate, examine and relate suitable information ...
- ... through text analysis, classification and understanding tasks
- So, Text Mining is not Information Access but relies on it
Applications I

- The same as KDD applications, but working with textual data
  - marketing
  - financial investment
  - decision support
  - science
- manufacturing
- health care
- fraud detection
- etc.

- We review some examples

Applications II

- Knowledge Management [Semio 02]
  - Enormous need to manage and control great quantities of textual and other information that drive businesses
  - Knowledge management is “…the process of capturing a company’s collective expertise wherever it resides—in databases, on paper, or in people's heads—and distributing it to wherever it can help produce the biggest payoff.”
Applications III

- For instance, TAKMI (Text Analysis and Knowledge Mining) by IBM [Nasukawa & Nagano 01]
- Mining textual databases in PC help centres, they can
  - automatically detect product failures
  - determine issues that have led to rapid increases in the number of calls and their underlying reasons
  - analyse help centre productivity and changes in customers’ behaviour involving a particular product

Applications IV

- Personal Intelligent Information Access Assistants [Mladenic 99]
- Help users to access (find, relate) information from several sources and to turn it into knowledge through, for instance
  - gathering and processing text
  - finding relations among pieces of information
  - providing suitable metaphors for information browsing
Applications V

• For instance, Personal WebWatcher [Mladenic 99]
  – A content-based intelligent agent that uses text-learning for user-customized web browsing
  – Note that the text writer may not be aware of some information consumers needs
  – For instance, the agent can be used to track competitors or clients web pages

Applications VI

• Predicting trends on the basis of textual evidence [Grobelnik et al. 00]
• For instance, EAnalyst [Lavrenko et al. 00]
  – Discovers the trends in time series of numeric data and attempts to characterize the content of textual data that precede these events
  – The objective is to use this information to predict the trends in the numeric data based on the content of textual documents that precede the trend
  – E.g. To predict the trend in stock prices based on the content of new articles published before that trend occurs
Problems with Textual Data I

- The known KDD problems and challenges [Fayyad et al. 96] extend to Textual Data
  - Large (textual) data collections
  - High dimensionality
  - Over fitting
  - Changing data and knowledge
  - Noisy data
  - Understand ability of mined patterns
  - etc...

Problems with Textual Data II

- But there are new problems
  - Text is not designed to be used by computers
  - Complex and poorly defined structure and semantics
  - But much harder, *ambiguity*
    - In speech, morphology, syntax, semantics, pragmatics
    - For instance, intentionality
  - Multilingualism
    - Lack of reliable and general translation tools
Text Mining and KDD: the process I

- We will focus on Text Mining techniques and subordinate text tasks
- But under the KDD standard process
- Text analysis, processing tasks play different roles in different steps

Text Mining and KDD: the process II

- The standard KDD process (borrowed from [Fayyad et al. 96])
Text Mining and KDD: the process III

- Information Retrieval
- Categorization
- Partial Parsing
- Summarization
- POS Tagging
- Term Clustering
- Transformation
- Data Mining
- Interpretation

Text Mining and KDD: the process IV

- Or the text-related task may be previous to a KDD process, as Information Extraction (IE) [Grobelnik et al. 00]
- IE aims at filling domain dependent templates with data buried in text items
- E.g. Web→KB aims at probabilistic, symbolic knowledge base that mirrors the content of the World Wide Web [Ghani et al. 00]
Content Based Text Processing Tasks I

- Taxonomy of Text Mining subtasks based on [Lewis 92]
- Dimensions
  - Size of text
  - Involve supervised or unsupervised learning
  - Text classification vs. understanding
    - Assigning documents or parts to a number of groups vs.
    - More complex access to document content
    - Note it is not a sharp division

Content Based Text Processing Tasks II

- Sample text classification tasks

<table>
<thead>
<tr>
<th></th>
<th>Words</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td>POS Tagging, Word Sense Disambiguation</td>
<td>Text Categorization, Filtering, Topic Detection and Tracking</td>
</tr>
<tr>
<td>learning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>learning</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Text Mining and Internet Content Filtering, ECML/PKDD Tutorial, August 19th, 2002 19

Text Mining and Internet Content Filtering, ECML/PKDD Tutorial, August 19th, 2002 20
Content Based Text Processing Tasks III

- Sample text understanding tasks

<table>
<thead>
<tr>
<th>Words</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised learning</td>
<td>Information Extraction</td>
</tr>
<tr>
<td>Unsupervised learning</td>
<td>Word Sense Discovery</td>
</tr>
</tbody>
</table>

Text Processing Basics I

- Most text processing tasks involve a number of steps, including
  - Text instances representation (indexing in Information Retrieval)
  - Learning
  - Evaluation and presentation
- Those steps can be seen as KDD steps over text data
  - For instance, text representation corresponds to selection, processing and transformation steps
Text Processing Basics II

- Text instances representation
  - The goal is to capture text semantics
  - A requirement is to avoid intensive manual processing (hand-coding) except for text labelling
  - Involves
    - Feature definition
    - Feature selection and extraction
      - This step reduces concept space dimensionality

Text Processing Basics III

- Feature definition
  - Text is usually represented as a “bag of words”
  - Which in fact is a the Information Retrieval (IR) Vector Space Model [Salton 89]
  - A text sequence is represented as a concept weight vector
  - Concepts are words, word stems, word phrases
  - Weights can be binary, frequency-based
Text Processing Basics IV

• Feature definition
  – Semantic similarity between natural language expressions is captured through the cosine formula

\[
\text{sim}(d_j, q_k) = \frac{\sum_{i=1}^{m} w_{dj} \cdot w_{qk}}{\sqrt{\sum_{i=1}^{m} w_{dj}^2 \cdot \sum_{i=1}^{m} w_{qk}^2}}
\]

Text Processing Basics V

• Feature definition
  – A very frequent representation involves
  – Filtering according to a high frequency (stop) word list to discard low-semantics words (prepositions, etc) [Salton 89]
    • BNC frequency lists, IR stop-lists
  – Stemming words to achieve a canonical concept representation (e.g. analysis, analysing, analyser are collapsed to ANALY)
    • Porter stemmer for English
Text Processing Basics VI

• Feature definition
  – Concept weights are often *tf.idf* kind [Salton 89]
    \[ W(i, j) = tf(i, j) \cdot \log_2 \left( \frac{N}{df(i)} \right) \]
  – *tf(i,j)* is the number of times that concept *i* occurs in document *j* of the text collection
  – *N* is the number of documents in the text collection
  – *df(i)* is the number of documents in which concept *i* occurs

Text Processing Basics VII

• Feature selection
  – A subset of the original concepts is extracted to avoid low representative concepts [Sebastiani 02]
  – If supervised learning follows, information theoretic or statistical measures are used to rank concepts according to their quality
  – Measures can be global (when quality for all classes is measured) or local (concepts are specific to each class)
Text Processing Basics VIII

- Feature selection
  - Some effective quality metrics include
    - Information Gain - IG (locally defined)
      \[ IG(i,k) = \sum_{x \in \{c_k, \bar{c}_k\}} \sum_{y \in \{t_i, \bar{t}_i\}} P(x,y) \cdot \log_2 \left( \frac{P(x,y)}{P(x) \cdot P(y)} \right) \]
      - Being \( t_i \) the \( i \)th concept and \( c_k \) the \( k \)th class in the text collection
      - Require text items labelled with class identifiers

Text Processing Basics IX

- Feature selection
  - Document Frequency (DF) is the number of documents in which the concept occurs
  - Does not require text items labelled with class identifiers
  - But DF can also be defined according to classes
    \[ DF(i,k) = P(t_i|c_k) \]
Text Processing Basics X

- Feature selection
  - Several more including odds ratio, $\chi^2$ [Sebastiani 02]
  - Variable effectiveness
  - For instance, for Text Categorization [Yang & Pedersen 97]
    - IG and $\chi^2$ are very effective (allow to eliminate 99% of concepts without effectiveness decrease in classification)
    - DF is quite effective (90% elimination)
    - Mutual Information and Term Strength are bad

Text Processing Basics XI

- Feature extraction
  - Based on the assumption that original concepts are not very representative (because of ambiguity, lack of statistical evidence)
  - A new concept set is produced by assorted procedures including
    - Thesaurus construction
    - Latent Semantic Indexing
    - Concept Indexing
    - Phrase construction
Text Processing Basics XII

- Feature extraction
  - Thesaurus construction [Salton 89]
    - Using co-occurrence statistics to detect semantic regularities across concepts
    - E.g. Class #764 in an engineering text collection is
      \[(\text{refusal}) \text{ refusal declining non-compliance rejection denial}\]
  - Also known as Term Clustering [Lewis 92]
  - Related to Latent Semantic Indexing and Concept Indexing

Text Processing Basics XIII

- Feature extraction
  - Latent Semantic Indexing [Deerwester et al. 90, Dumais 95]
    - A set of document vectors indexed according to a set of concepts is transformed to reduce the number of concept dimensions
    - A mapping function is obtained by applying a singular value decomposition to the matrix formed by the original document vectors
  - Address synonymy and polysemy
Text Processing Basics XIV

• Feature extraction
  – Concept Indexing
  – By using a semantic net or ontology of concepts
  – For instance, the Lexical Database WordNet in [Gonzalo et al. 98]
  – Faces a problem of ambiguity
  – An automatic, effective Word Sense Disambiguation process is required
  – But may be language independent using EuroWordNet [Vossen 98]

Text Processing Basics XV

• Feature extraction
  – Phrase construction
    • Concepts sequences can be recorded (n-grams) and added to the concept set
    • Alternatively, some Natural Language Processing can be applied to build linguistically motivated phrases (noun phrases)
      – For instance, using Part-Of-Speech Tagging, shallow parsing and regular expressions
    • Mixed results [Lewis 92, Riloff & Lehnert 94]
Text Processing Basics XVI

• Learning
  – Supervised learning algorithms
    • Most are general Machine Learning (ML) algorithms including
decision tree learners, rule learners, neural networks and
Support Vector Machines (SVM), (Naive) Bayesian
approaches, linear function learners, instance-based
classification, etc
    • Some specific algorithms like Rocchio from IR
    • Maybe the most effective are SVMs
  – Unsupervised learning algorithms
    • Again most coming from ML including hierarchical clustering
methods, Expectation-Maximization, k-means, etc

Text Processing Basics XVII

• Direct evaluation
  – Supervised classification
    • Standard metrics coming from IR and ML including
Recall, Precision, Accuracy, Error, $F_\beta$, etc
    • Statistical tests not often used
  – Unsupervised classification
    • Comparing with a manual classification
    • Entropy, etc
Text Processing Basics XVIII

- Direct evaluation in Supervised classification

\[
\begin{array}{c|c|c|c|c}
  \text{System} & \text{C} & \text{¬C} \\
  \hline
  \text{C} & \text{tp} & \text{fp} \\
  \text{¬C} & \text{fn} & \text{tn} \\
\end{array}
\]

**Contingency matrix**

Recall \( r = \frac{tp}{tp + fn} \)

Precision \( p = \frac{tp}{tp + fp} \)

Accuracy \( = \frac{tp + tn}{tp + fn + fp + tn} \)

\[
F_{\beta} = \frac{1}{\beta \frac{1}{p} + (1 - \beta) \frac{1}{r}}
\]

\[
F_1 = \frac{2pr}{p + r}
\]

Text Processing Basics XIX

- Indirect evaluation
  - The task A is a part of a more complex task B
  - Approaches for task A are compares as they affect task B effectiveness
  - For instance, a Word Sense Disambiguation approach can be evaluated as it affects an automatic translation approach
Text Processing Basics XX

• Visualization
  – From standard visualization tools from KDD...
    • Including decision tree graph tools, etc
  – To specific metaphors designed for text presentation (in Human-Computer Interaction, Information Access, etc)
    • Including those by Hearst and others
  – In Hearst opinion, visualization is the core of Text Mining systems [Hearst 99], because TM is user/application centric

Text Processing Basics XXI

• Decision Tree in Mine Set by Silicon Graphics
Text Processing Basics XXII

- Clusters presentation in ThemeScapes [Wise et al.95]

Text Processing Basics XXIII

- Hierarchical presentation in Cat-A-Cone [Hearst & Karadi 97]
Text Processing Basics XXIV

- Hypertext graphics in Mapuccino (formerly WebCutter) [Maarek & Shaul 97]

Text Processing Basics XXV

- Query word frequency in TileBars [Hearst 95]
Case Study: CORA I

- CORA is a publicly available search engine on computer science research papers [McCallum et al. 00]
- Available at http://www.cora.whizbang.com/
- Built using a number of text processing techniques
- It can be used for computer scientific knowledge discovery

Case Study: CORA II
Case Study: CORA III

- Integrates a number of techniques and tasks
  - Intelligent spidering the web for computer science research papers using reinforcement learning
  - Text Categorization to automatically classify documents into a topic hierarchy, using Naive Bayes and Expectation-Maximization
  - Information Extraction for the identification of titles, authors, etc using Hidden Markov Models
  - Information Retrieval for accessing documents and citation analysis for ranking according impact

Summarizing

- We will review a number of content based text processing tasks
- The goals are
  - to describe them and introduce their techniques
  - to show their role in Text Mining
  - to analyse one of them (text categorization) as a KDD process itself
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

2
LEARNING WHEN KNOWING
Outline

1. Introduction
2. Text Categorization
3. Text Filtering
4. Topic Detection and Tracking
5. Part Of Speech Tagging
6. Word Sense Disambiguation
7. Shallow Parsing
8. Information Extraction

Introduction

• Supervised learning tasks
  – Textual data manually labelled with class values
  – Possibility of using concept quality metrics in text representation
  – Automatic induction of automatic classifiers by Machine Learning (ML) algorithms
  – Hit-mistake evaluation metrics (precision, recall, etc)
Text Categorization I

- Text categorization (TC) is the automatic assignment of documents to a set of predefined classes
- Classes are usually content based (topics, keywords, subject headings) but can also be genres, authors, etc
- Documents are e-mails, reports, books, web pages, news items, etc

Text Categorization II

- The dominant approach is
  - Given a set of manually classified (labelled) documents
  - Use IR and ML techniques to induce an automatic classifier or new documents
- This way, the knowledge acquisition of knowledge based classifiers is alleviated
- See [Sebastiani 02] for an in-depth survey
Text Categorization III

- Applications
  - Knowledge management (automatic document organization for knowledge sharing)
  - Document indexing in libraries
  - Web page classification into Internet directories
  - We will later focus on harmful Internet document identification for filtering (spam e-mail, pornographic web pages)
  - Many others including author identification, automatic essay grading, language guessing, etc

Text Categorization IV

- The basic model involves
  - Documents content representation as bags of words with stop words filtering and word stemming
  - Feature selection and extraction
  - Learning a classifier using some ML algorithm including the full range of options
  - Evaluating the effectiveness of the learned classifier
Text Categorization V

• Document representation
  – Riloff describes experiments in which stemming negatively affects performance [Riloff 95]
  – Document structure is rarely used, except for hypertext
    • For instance, in [Attardi et al. 99] HTML documents are represented by their blurb (hyper linked text pointing to the document)
    • Also, web pages can be classified using only hyperlink net structure, detecting hubs and authorities [Chakrabarti et al. 98]
Text Categorization VII

• Feature selection
  – The most basic approach is deleting low frequency terms
  – A wide range of statistical quality metrics
    • Concepts are selected according their predictive value
    • Include IG, $\chi^2$, DF, etc discussed above
  – The most effective are IG, $\chi^2$ and DF according to [Yang & Pedersen 97]

Text Categorization VIII

• Feature selection
  – Interestingly, class dependent metrics can be averaged over all classes
  – Given a metric denoted by $X(t,c)$, being $t$ a concept and $c$ a class in a set $C$, several possible averages including
    
    $$X_{\text{avg}}(t) = \sum_{c \in C} P(c)X(t,c)$$
    $$X_{\text{max}}(t) = \max_{c \in C}\{X(t,c)\}$$
Text Categorization IX

• Feature extraction
  – Latent Semantic Indexing can be considered a positive technique (see [Sebastiani 02])
  – Concept indexing based on taxonomies like the lexical database WordNet is
    • Successful for IR (e.g. [Gonzalo et al. 98])
    • Unsuccessful for TC [Junker & Abecker 97, Scott & Matwin 99]
    • But WordNet can be successfully used in TC [Buenaga et al. 00, Ureñña et al. 01, Benkhalifa et al. 01]

Text Categorization X

• Machine learning classifiers
  – Nearly all methods and algorithms have been applied to the task
  – Most effective approaches include
    • Support Vector Machines (e.g. [Dumais et al. 98, Drucker et al. 99])
    • K-Nearest Neighbours (e.g. [Yang 99, Larkey 99])
    • AdaBoost-ed C4.5 (e.g. [Schapire & Singer 00])
Text Categorization XI

- Machine learning classifiers
  - Support Vector Machines
  - The goal is finding a surface that separates the positives from the negatives by the widest possible margin
  - The SVM method chooses the middle element from the “widest” set of parallel hyper planes in the N-dimensional space (being N the number of indexing concepts)

Text Categorization XII

- Machine learning classifiers
  - Support Vector Machines
  - Positive (+) and negative (o) instances
  - 2-dimensional space
  - Detecting the most important instances for separating rest of examples
  - Called support machines
  - Instances need not to be linearly-separable
  - Separating surfaces need not to be hyper planes

Borrowed from [Sebastiani 02]
Text Categorization XIII

• Machine learning classifiers
  – Support Vector Machines
  – Good effectiveness
  – Fast training for linear SVM (which result in linear classification functions)
  – Feature reduction not required
    • Robust to over fitting

Text Categorization XIV

• Machine learning classifiers
  – k-Nearest Neighbours
  – A king of example/instance based classification, or memory based learning
  – Classifying a new instance using the classes of known instances
  – Voting classes of k neighbours according to distance to the new instance
  – Several distances available (cosine, dot product, Euclidean)
Text Categorization XV

- Machine learning classifiers
  - k-Nearest Neighbours

Comparison with centroid-based classification, borrowed from [Sebastiani 02]

Text Categorization XVI

- Machine learning classifiers
  - k-Nearest Neighbours
  - Very effective in classification
  - Not very efficient (but indexing techniques are now web scale, see Google)
Text Categorization XVII

- Machine learning classifiers
  - Boosting
  - Combining a set (committee) of same-kind classifiers successively learned by a weaker method (e.g. C4.5)
  - Next classifier is induced mainly on instances misclassified by previous classifiers
  - Classification is based on weighted vote of all learned classifiers
  - Good results in BOOSTEXTER (AdaBoost+C4.5)

Text Categorization XVIII

- Evaluation
  - Mainly concerned with effectiveness, less with efficiency
  - Standard IR & ML metrics presented above (recall, precision, accuracy, $F_1$, etc)
  - In multi class situations, at least report $F_1$ by
    - Macro averaging – averaging on the number of classes
    - Micro averaging – computing over all decisions at once
Text Categorization XIX

- Evaluation
  - Cross-validation is not frequent
  - Some available test collections include
    - Reuters-21578
    - The Reuters Corpus Volume 1
    - OHSUMED
    - 20-NewsGroups
    - Ling-Spam

Text Categorization XX

- Evaluation
  - Scarce statistical testing (intro in [Yang & Liu 99])
  - Accuracy and error do not fit well TC because class distribution is usually highly biased
  - Now an increasing use of cost-sensitive metrics for specific tasks (e.g. weighted accuracy, ROCCH method [Gomez 02])
Text Categorization XXI

• Interesting work in TC
  – Using unlabelled data in TC (e.g. [Nigam et al. 00] )
  – Using Yahoo-like hierarchical structure (e.g. [Mladenic 98])
  – Using other text features (e.g. [Forsyth 99, Kessler et al. 97, Sahami et al. 98, Gómez et al. 00])
    • Linguistic-like in genre or author identification, spam classification

Text Filtering I

• Text Filtering (TF) is an information seeking process in which documents are selected from a dynamic text stream to satisfy a relatively stable and specific information need
• E.g. News items from newspapers are daily collected and delivered to a user according his/her interests = the personalized newspaper
• See e.g. [Oard & Marchionini 96] for overview
Text Filtering II

- Also known as Selective Delivery of Information (SDI)
- Systems use IR, ML and User Modelling techniques to induce and refine a user model which is used to select new documents from the stream
- Collaborative vs. content based filtering

Text Filtering III

- Product recommendation in Amazon (content)
  - According a personal profile accounting for
    - A set of categories (DVD, Computer games, Music) and subcategories (genres)
  - Starting with preferred items
    - Authors, titles, brands
  - Recommendation of new releases
  - Of course it is not text-content based, but on the purchasing history
Text Filtering IV

- Product recommendation in Amazon (collaborative or social)
  - According to other customers purchases
    "Customers who bought this book also bought..."
  - Based on
    • previous annotations by other users
    • and generating a user segmentation
- A trend is combining both ideas (see e.g. [Good et al. 99])

Text Filtering V

- Applications
  - CRM & marketing (e.g. cross-selling, recommendation)
  - Information delivery at organizations for Knowledge Management
  - Information access assistants (Personal WebWatcher [Mladenic 99])
  - Filtering news items in Newsgroups
  - In Text Mining, allows to personalize information access to the current knowledge discovery task
Text Filtering VI

• The process in content-based filtering
  – The filter usually starts at least with
    • A set of selected documents or
    • A set of user-defined keywords
  – A basic user model is induced from that information
  – The user model is refined according to relevance judgements from the user

Text Filtering VII

• The basic model involves
  – Documents content representation as bags of words with stop words filtering and word stemming
  – Feature selection
  – Learning and updating a classifier usually from relevance feedback in IR
  – Evaluating the effectiveness of the learned classifier
Text Filtering VIII

• Following [Allan 96]
  – The user initially describes his/her information interest as a standard keyword-based query
  – Documents come in batches, and after each arrival the user provides relevance judgements
  – The user model is refined using judged documents, but
    • It is assumed the system cannot store all documents permanently
    • Original user concepts must be retained to avoid erroneous bias

Text Filtering IX

• Following [Allan 96]
  – After each batch is judged, 100 top scoring concepts are added for the next cycle
    • Concepts in documents from the new batch are first ordered by rtf (# times they occur in positive documents)
    • Top 500 concepts are re-ranked according to the Rocchio formula
      \[
      \text{Rocchio}_i = w_{\text{query}} + 2w_{\text{rel}} - \frac{1}{2}w_{\text{non-rel}}
      \]
    • Where weights (w) are computed in a tf.idf fashion
    • And top 100 terms are added (always retaining user’s)
Text Filtering X

• Following [Allan 96]
  – A precision-based evaluation shows that adding concepts in batches is nearly as effective as adding all concepts at once
  – Note that concepts are never removed from the profile
  – Other representations and algorithms are possible
    • E.g. [Bloerdon et al. 96] use thesaurus categories as concepts
    • E.g. [Tan & Teo 98] use a neural-network-kind algorithm

Topic Detection and Tracking I

• Topic Detection and Tracking (TDT) is the automatic detection of novel events in chronologically-ordered streams of news stories, and tracking these events over time
• Events the user want to detect are not know previously by him/her, and so retrieval is inadequate
• TDT is an application by itself
• See [Yang et al. 99] for an overview
**Topic Detection and Tracking II**

- Events (USAir-417 crash) are not topics (airplane accidents) and must be separated
- TDT is probably the purest Text Mining task since we want to discover new events
- TDT approaches combine several kinds of learning including
  - supervised (TC) for tracking
  - and unsupervised (Clustering) for detection

**POS-Tagging I**

- Part Of Speech Tagging (POS-Tagging) is labelling each word in a sentence with its appropriate part of speech
  - E.g. The-AT representative-NN put-VBD chairs-NNS on-IN the-AT table-NN
  - Where AT = Determiner-article, NN = Noun-singular, VBD = Verb-past tense, NNS = Noun-plural, IN = Preposition
- Words show limited POS ambiguity
- See [Manning & Schütze 99] for an overview
POS-Tagging II

- There are several tag sets (Brown tag set, Penn Treebank tag set, etc) ranging in granularity, complexity, etc
  - E.g. 56 tags in the Penn Treebank
  - E.g. 197 tags in the London-Lung Corpus
- Granularity directly affects performance

POS-Tagging III

- POS-Tagging makes sense as an intermediate task for others
- E.g. with shallow parsing
  - For creating linguistically motivated indexing terms (retrieval, categorization, filtering, etc)
  - For detecting slot fillers candidates in Information Extraction
  - For detecting answer candidates in Question Answering
POS-Tagging IV

• A range of approaches
  – Markov Models (e.g. [Church 88, DeRose 88])
  – Hidden Markov Models (e.g. [Jelinek 85, Cutting et al. 92a]) => the most popular
  – Transformation-based learning [Brill 95]
  – Decision trees [Màrquez et al. 00, Schmid 94]
  – Etc

Word Sense Disambiguation I

• Most words in natural language have several meanings or senses
• E.G. Bank in WordNet
  1. depository financial institution, bank, banking concern, banking company -- (a financial institution that accepts deposits and channels the money into lending activities; "he cashed a check at the bank"; "that bank holds the mortgage on my home")
  2. bank -- (sloping land (especially the slope beside a body of water); "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents")
  ...
  10. bank -- (a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning))
Word Sense Disambiguation II

• Word Sense Disambiguation (WSD) is to determine which of the senses of an ambiguous word is invoked in a particular use of the word
• Usually, WSD is stated as
  – Having a set of word senses candidates listed or defined in a dictionary, thesaurus, etc
  – Detecting the most suitable sense of a word among them given the context of usage
• Good overview in [Manning & Shütze 99], Chapter 7

Word Sense Disambiguation III

• Again, WSD makes sense as an intermediate task for other text processing tasks, including
  – Machine translation (because each sense may result in a different translation)
  – IR and TC (because the sense of a word is highly influential in the relevance or adequacy as predictor of the word)
  – Spelling correction (for instance, to determine when diacritics should be inserted)
Word Sense Disambiguation IV

- Two kind of methods
  - Dictionary based (running from [Lesk 86] and [Yarowsky 92] to [Agirre & Rigau 96])
  - Training corpus based (from [Gale 92] and [Mooney 96] to [Pedersen 02])
- A trend is the combination of techniques (e.g [Stevenson & Wilks 99])

Word Sense Disambiguation V

- Dictionary based WSD
  - Information sources are
    - The text context of occurrence of the word to disambiguate
    - The available information for the senses in the dictionary
  - Hence, it is no supervised
  - The basic approach is comparing (e.g. computing the overlap) between both information sources
Word Sense Disambiguation VI

- Dictionary based WSD
  - A very interesting approach is that in [Agirre & Rigau 96]
  - Information about senses is collected from WordNet considering semantic relations and population of the hierarchy
  - The system tries to resolve the lexical ambiguity of nouns by finding the combination of senses from a set of contiguous nouns that maximises the Conceptual Density among senses

Word Sense Disambiguation VII

- Training based WSD
  - Information sources are
    - The text context of occurrence of the word to disambiguate
    - The senses in the dictionary
    - A manually disambiguated corpus (e.g. Semcor)
  - A supervised learning based classifier is trained on the corpus to enable predictions on new words
Word Sense Disambiguation VIII

- Training based WSD
  - An interesting study is that in [Mooney 96]
  - The task is disambiguating the word “line” manually tagged according to its six Wordnet senses
  - For instances representation, the words occurring in the precedent and current sentences, filtered according a stoplist and further stemmed are taken as binary features

Word Sense Disambiguation IX

- Training based WSD [Mooney 96]
  - A number of learning approaches are compared including a Bayesian classifier, a perceptron, a decision tree learner, kNN, two rule learners and a decision list learner
  - The training data is highly biased to the “product” sense (it occurs more that five times than others)
  - Bayes and the perceptron perform best according to precision
Shallow Parsing I

• Also named Robust Parsing, Chunk Parsing and Chunking [Abney 91, Vergne 00, Tjong & Buchholz 00]
• Given a sentence, the goal is to find a partial parsing of it, in which non-overlapping phrases and the relations among them are identified
• E.g.
  “He reckons the current account deficit will narrow to only # 1.8 billion in September.”
  [NP He ] [VP reckons ] [NP the current account deficit ] [VP will narrow ] [PP to ] [NP only # 1.8 billion ] [PP in ] [NP September ]

Shallow Parsing II

• Very related to POS Tagging
  – POS Tagging is nearly a requirement for chunking
  – Sometimes, the same methods are used for both tasks (e.g. Hidden Markov Models)
  – Both tasks are syntactic annotation at very close levels of complexity (neither of them capture natural language recursive nature)
  – In fact, they are combined for the applications, that includxe sophisticate indexing for text classification and finding slot fillers in Information Extraction
Shallow Parsing III

- It is worth studying the chunking evaluation at the Computational Natural Language Learning Workshop 2000
- The Penn Treebank was processed to convert full parses into chunk sequences
- There were 211727 tokens and 106978 chunks, where 55081 (51%) were noun phrases, 21467 (20%) were verb phrases and 21281 (20%) were prepositional phrases

Shallow Parsing IV

- A number of approaches were tested
  - Rule based systems hand coded from scratch or adapted from a full parser (3)
  - Memory based systems (1)
  - Statistical methods including Markov Models, Hidden Markov Models and maximum entropy methods (4)
  - Combined approaches with committees built over a variety of base learners (SVMs, memory based, etc)
Shallow Parsing V

<table>
<thead>
<tr>
<th>Approach</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combined SVM</td>
<td>93.45%</td>
<td>93.51%</td>
<td>93.48</td>
</tr>
<tr>
<td>Combined WPDV &amp; MBL</td>
<td>93.13%</td>
<td>93.51%</td>
<td>93.32</td>
</tr>
<tr>
<td>Combined MBL</td>
<td>94.04%</td>
<td>91.00%</td>
<td>92.50</td>
</tr>
<tr>
<td>Hidden Markov Models</td>
<td>91.99%</td>
<td>92.25%</td>
<td>92.12</td>
</tr>
<tr>
<td>Rules</td>
<td>91.87%</td>
<td>92.31%</td>
<td>92.09</td>
</tr>
<tr>
<td>Maximum Entropy</td>
<td>92.08%</td>
<td>91.86%</td>
<td>91.97</td>
</tr>
<tr>
<td>Maximum Entropy</td>
<td>91.65%</td>
<td>92.23%</td>
<td>91.94</td>
</tr>
<tr>
<td>Memory Based (MBL)</td>
<td>91.05%</td>
<td>92.03%</td>
<td>91.54</td>
</tr>
<tr>
<td>Markov Models</td>
<td>90.63%</td>
<td>89.65%</td>
<td>90.14</td>
</tr>
<tr>
<td>Rules</td>
<td>86.24%</td>
<td>88.25%</td>
<td>87.23</td>
</tr>
<tr>
<td>Rules</td>
<td>88.82%</td>
<td>82.91%</td>
<td>85.76</td>
</tr>
<tr>
<td>baseline</td>
<td>72.58%</td>
<td>82.14%</td>
<td>77.07</td>
</tr>
</tbody>
</table>

Shallow Parsing VI

- Best results for combined methods and specially a dynamic programming combination of SVMs
- With these results
  - It is possible to accurately detect noun phrases for indexing in text classification systems
  - But probably more precision is required for Information extraction
Information Extraction I

- The goal of Information extraction (IE) is transform text into a structured format (e.g. database records) according to its content
  - E.g. Heterogeneous researchers homepages are transformed into database records containing name, position, institution, research interests, projects, etc
  - E.g. Terrorism news articles are transformed into records including kind of incident, place, date, instigator, personal damages, etc

Information Extraction II

- A key application is feeding other text and mining processes (see e.g. [Nahm & Mooney 02] about the project DISCOTEX)
- Also structured databases are given to analysts to support their work, e.g. finding trends and forecasting according to them
- Introduction regarding web content [Eikvill 99]
- See [Cowie & Lehnert 96, Cunningham 99]
Information Extraction III

- Techniques range from knowledge poor to rich, with obvious increasing domain dependence and effectiveness
- Approaches depend on the structure of text
  - Free text with fully grammatical sentences allow natural language processing techniques with the induction of patterns based on syntactic and semantic analysis
  - ...

Information Extraction IV

- ...
- Structured text follows a predefined and structured format that leads to delimiter based patterns
- Semi-structured text is telegraphic and ungrammatical, thus a combination of techniques that employ several sources of information are used
Information Extraction V

- A popular approach is the automatic induction or manual derivation of *wrappers* (see e.g. [Freitag & Kushmerick 00])
- A wrapper is a procedure for extracting a particular resource's content
- Usually consists of a set of extraction rules and a rule engine
- A wrapper is information source dependent

Information Extraction VI

- Wrapper construction can be done through inductive learning, by reasoning about a sample of the resource's documents
- Kushmerick et al. have identified several classes of wrappers which are *reasonably useful, yet efficiently learnable*
- To assess usefulness, they measured the fraction of Internet resources that can be handled by their techniques and found that their system can learn wrappers for 70% of the surveyed sites
Summary

- In this track we have presented a sample of tasks and techniques that
  - Are oriented to supervised learning from text
  - In the context of TM
- In a “real” TM environment (in the sense by Hearst), the tasks are successively applied to texts, and combined with unsupervised tasks and techniques
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

3

LEARNING WHEN NOT KNOWING
Outline

1. Introduction
2. Document Clustering
3. Term Clustering
4. Document Summarization

Introduction

• Unsupervised learning tasks
  – Unlabelled textual data
  – Limited concept quality metrics in dimensionality reduction
  – Unsupervised induction of text groups through data clustering algorithms
  – Coherence or task-dependent evaluation metrics
• More strict TM tasks
Document Clustering I

• Document Clustering (DC) is partitioning a set of documents into groups or clusters
• Clusters should be computed to
  – Contain similar documents
  – Separate as much as possible different documents
• For instance, if similarity between documents is defined to capture semantic relatedness, documents in a cluster should deal with the same topics, and topics in each cluster should be different

Document Clustering II

• DC applications include
  – Exploratory text data analysis
    • In words by Manning & Schütze [99]
      “It is always a mistake to not first spend some time getting a feel for what the data at hand look like” (p. 497)
  – Pre-processing for other tasks, e.g.
    • In [Karypis & Han 00], to detect the main semantic dimensions of the text collection and to support a kind of Concept Indexing
    • In [Hatzivassiloglou et al. 01] for text summarization
Document Clustering III

• But the main application is supporting a variety of Information Access tasks
  – Guiding the organization of a document collection (e.g. [Sahami 98])
    • Progressively clustering groups of documents allow to build a topic hierarchy
  – Supporting browsing and interactive retrieval (e.g. [Cutting et al. 92b, Baldonado & Winograd 97, Wu et al. 01]), now some search engines (Vivisimo)
    • Grouping retrieved documents to allow a faster relevant documents selection process

Document Clustering IV

• As other text processing tasks, DC has several steps
  – Document representation
  – Dimensionality reduction
  – Applying a clustering algorithm
  – Evaluating the effectiveness of the process
• Basically, we can apply the same methods that we use in data clustering (see the survey by Jain et al. [99])
Document Clustering V

• Document representation
  – Again, documents are represented as concept weight vectors
  – Concepts are usually words filtered according stop lists and stemmed
  – Weighting approaches include binary, $tf$, $tf.idf$, etc

Document Clustering VI

• Dimensionality reduction
  – Two main approaches, depending on the knowledge used
    • If we have not knowledge, distributational approaches are often applied: the Zipf’s law
      – Very frequent and very unusual terms are filtered out
      – They have less discriminative power (see [Salton & McGill 83; Salton 89])
    • If we have knowledge (e.g. we have query concepts in interactive retrieval), a number of heuristics are used
      (see e.g. [Rüger & Gauch 00])
Document Clustering VII

- Dimensionality reduction
  - The Zipf's law and discriminative power

![Discriminative power vs frequency](image)

- Most discriminative concepts have low to medium frequency

Document Clustering VIII

- Clustering algorithms
  - Include a wide sample of the available in all required conditions
    - Hierarchical versus flat
    - Hard versus soft
    - Several semantic similarity functions
  - Perhaps the most often applied method is a kind of Hierarchical Agglomerative Clustering (HAC) method, but sometimes Expectation-Maximization (AutoClass) and Self-Organizing Maps
Document Clustering IX

- HAC (as in [Manning & Schütze 99])
  - Starts with a cluster per document
  - In each iteration, the closest pair of clusters are merged
  - It depends on the way similarity between documents and between clusters is defined
    - Typical inter document similarity metrics are cosine, Euclidean, etc
    - Similarity between clusters can be measured by single link, complete link, group-average, etc methods

Document Clustering X

- EM (as in [Manning & Schütze 99])
  - Can be seen as a way of estimating the hidden parameters of a model
    - Given some data X, and a model M with $\theta$ parameters, we want to estimate $P(X|M(\theta))$ and to find the model that best fits the data (maximizes the likelihood of the data)
  - Beginning with an approximation of $\theta$, iterates two steps
    - In the expectation step, the parameters of the model are estimated and interpreted as cluster membership probabilities
    - In the maximization step, the most likely parameters are estimated given the cluster membership probabilities
Document Clustering XI

• DC quality metrics
  – Sometimes, the results are compared to a manual classification taken as golden standard leading to accuracy based metrics (entropy, $F_B$, Mutual Information, etc) (see e.g. [Zhao & Karypis 02, Vaithyanathan & Dom 99])
  – With no information of class labels, metrics like overall similarity (cohesiveness) are used (e.g. [Steinbach et al. 00])

Document Clustering XII

• DC quality metrics
  – Also indirect evaluation (in the context of a second task) is possible
  – For instance, the increase of retrieval effectiveness in a text retrieval experiment including
    • Direct retrieval effectiveness metrics (recall, precision, etc) [Leuski 01]
    • Time to find the information [Hatzivassiloglou et al. 01]
Term Clustering I

• This is an umbrella that includes a number of tasks and techniques, e.g
  – Automatic thesaurus construction
  – Latent semantic indexing
• The basic idea is to work at the word level to develop models usable in indexing for other document level tasks

Term Clustering II

• Automatic Thesaurus Construction (ATC)
  – A thesaurus is (traditionally) a dictionary of synonyms, but the concept has evolved to include semantic relations between concepts
    • Class #764 of an engineering related thesaurus
      (refusal) refusal declining non-compliance rejection denial
  – A text processing oriented thesaurus contains
    • Concepts that contain words and multi-word expressions
    • Relations like is-a and has-a
  – In fact, semantic nets
Term Clustering III

• Automatic Thesaurus Construction
  – Research in ATC begins with IR (50’s) (see consolidated references as Salton’s)
  – The problem of lack of statistics for individual but related words is addressed by grouping semantically related words into classes
  – The classes in the thesaurus are after used for better representing document contents

Term Clustering IV

• Automatic Thesaurus Construction
  – It is possible to use the same techniques as those for document clustering
  – Traditional IR techniques are based on
    • The hypothesis that related words occur in similar contexts
    • So the similarity between words is computed through the similarity between documents in which occur
Term Clustering V

- **Automatic Thesaurus Construction**
  - So we work on a document x concept matrix
  - We compute similarity between concepts as similarity between the concept columns
  - Similarity between clusters is computed also using single link, complete link, group-average, etc approaches

Term Clustering VI

- **Automatic Thesaurus Construction**
  - Similarity between concepts may be computed through the cosine formula
  
  \[
  \text{sim}(k,l) = \frac{\sum_{j=1}^{c} \text{wd}_{kj} \cdot \text{wd}_{lj}}{\sqrt{\sum_{j=1}^{c} \text{wd}_{kj}^2 \cdot \sum_{j=1}^{c} \text{wd}_{lj}^2}}
  \]

  - Where \( \text{wd}_{kj} \) and \( \text{wd}_{lj} \) are the weights of the kth and lth concepts in the jth document of the collection
Term Clustering VII

- Automatic Thesaurus Construction
  - Again a variety of clustering methods can be used, being HAC the most frequent
  - While intuition supports that using semantic classes should lead to better representation, experience in IR is negative [Salton & McGill 83, Salton 89]
  - For instance, a relevance feedback iteration can be 10 times better than it

Term Clustering VIII

- Latent Semantic Indexing
  - It has been presented as a way to capture the main semantic dimensions in a text collection, avoiding synonymy and polysemy problems
  - Exploits co-occurrence information between concepts to derive a text representation based on new, less dimensions
  - Can be seen as an effective dimensionality reduction method
Term Clustering IX

• Latent Semantic Indexing
  – Conceived for IR [Deerwester et al. 90, Berry et al. 95]
  – E.g. applied to
    • TC [Dumais & Nielsen 92]
    • Filtering [Dumais 95]
    • Cross-Language IR [Dumais et al. 97]

Term Clustering X

• Latent Semantic Indexing
  – E.g. (borrowed from [Manning & Schütze 99])
  Let $A$ a concept x doc matrix for a text collection

$$\begin{pmatrix}
  d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\
  \text{cosmonaut} & 1 & 0 & 1 & 0 & 0 & 0 \\
  \text{astronaut} & 0 & 1 & 0 & 0 & 0 & 0 \\
  \text{moon} & 1 & 1 & 0 & 0 & 0 & 0 \\
  \text{car} & 1 & 0 & 0 & 1 & 1 & 0 \\
  \text{truck} & 0 & 0 & 0 & 1 & 0 & 1
\end{pmatrix}$$
Term Clustering XI

- Latent Semantic Indexing
  - E.g. (borrowed from [Manning & Schütze 99])
  - Given A, it can be observed that
    - Cosmonaut & astronaut are synonyms and never co-occur, but they do with moon
    - sim(d₂, d₃) = 0 but they contain synonym concepts
    - sim(d₅, d₆) = 0 but they contain synonym concepts
  - Of course, there are two main semantic dimensions in the data (astronomy, road)

Term Clustering XII

- Latent Semantic Indexing
  - E.g. (borrowed from [Manning & Schütze 99])
  - After LSI has applied, reducing to 2 dimensions
Term Clustering XIII

• Latent Semantic Indexing
  – The basic idea is mapping a high-dimensional space into a low-dimensional one
  – Iteratively choosing dimensions corresponding to the axes of greater variation
  – Co-occurring concepts are mapped onto the same dimension
  – A method called Singular Value Decomposition for the analysis of co-occurrence patterns is the core

Document Summarization I

• Document Summarization is the task of abstracting key content from one or more information sources
• It can be seen as a classification (knowledge poor) or understanding (knowledge rich) task
• Interesting overview in [Hahn & Mani 00]
Document Summarization II

- Mostly applied to ease information access
  - E.g. Most useful keywords are extracted from a set of documents (e.g. a cluster) to describe it
  - E.g. Documents in a collection are abstracted to avoid reading the full content
  - E.g. Documents retrieved from search are summarized to allow the user a faster identification of those relevant to the query

Document Summarization III

- We will classify approaches by size of the text unit used in the summary
  - Keyword summaries
  - Sentence summaries
- Although many classification based techniques for sentence summaries can be applied to keyword summaries
Document Summarization IV

• Keyword summaries
  – Abstracting is equivalent to detect most informative keywords or multi word expressions in a (set of) document(s)
  – For one document, it can be as simple as selecting the higher weight concepts in it (according to $tf.idf$)
  – In a interactive retrieval setting
    • should make the document as different as possible to other non related documents
    • should explain why the document has been retrieved

Document Summarization V

• Sentence summaries (according to [Hahn & Mani 00])
• Two taxonomies
  – Regarding function
    • Indicative
    • Informative
    • Critical
  – Regarding target user
    • Generic
    • Used-focused
Document Summarization VI

- Sentence summaries [Hahn & Mani 00]
  - The basic steps are
    - Analysing the source text
    - Determining its salient points
    - Synthesizing an appropriate output
  - We will focus on how knowledge poor (classification based) summarizing systems address these steps

Document Summarization VII

- Sentence summaries [Hahn & Mani 00]
  - Text units are sentences
  - A linear model of salience is often applied using a set of features (heuristics)
    - Location in the source text
    - Appearance of cue phrases
    - Statistical significance
    - Additional information
    \[ \text{Weight}(U) = \text{Location}(U) + \text{CuePhrase}(U) + \text{StatTerm}(U) + \text{AddTerm}(U) \]
Document Summarization VIII

• Sentence summaries [Hahn & Mani 00]
  – Location criterion
    • Weight sentences according to the part of the paragraph or document they occur in
      – Most news stories begin with a small summary
      – Sentences in the introduction and conclusions of research papers are very likely to occur in the summary
  – Cue Phrase criterion
    • Lexical or phrasal summaries as “in conclusion”, “in this paper”, etc
    • The approach suggest to overweight the sentences in which they occur

Document Summarization IX

• Sentence summaries [Hahn & Mani 00]
  – Statistical salience
    • Well known IR heuristic weights as tf.idf applied to select those sentences in which the concept occur
  – Additional term
    • Depending on the application, we can
      – Promote sentences that include query concepts (retrieval => query biased summaries)
      – Promote sentences that include user profile concepts (filtering => used adapted summaries)
Document Summarization X

- Sentence summaries
  - In [Kupiec et al. 95]
    - A supervised learning approach has been devised (and applied to the paper itself)
    - It has been shown that location combined with cue phrases is a very powerful method
  - In [Maña et al. 99]
    - Query biased summaries demonstrate their informativeness in a relevance feedback process

Document Summarization XI

Process in a supervised learning based summarizer [Hahn & Mani 00]
Document Summarization XII

• Sentence summaries
  – Evaluation
    • Intrinsic evaluation using human judgements (gold standard)
    • Extrinsic evaluation
      – The summarizer is as good as it contributes to a task in which is applied
      – For instance, the informativeness of the summary can be measured in terms of the accuracy of a summary based retrieval in comparison with a full-document retrieval

Document Summarization XIII

• A note about knowledge-rich summarization
  – A very common approach is to fill a template with facts in the text and after producing a canned-text summary
    • Filling the template is a IE task
    • Domain dependent
  – But more complex systems have been devised, including the attempt to capture meaning and appropriate output planning (e.g. SUMMARIST [Hovy & Lin 99])
Summary

• In this track we have presented a sample of tasks and techniques that
  – Are mostly oriented to unsupervised learning from text in the context of TM
  – Specially clustering-based problems are closer to real TM
• Again, in a “real” TM environment (in the sense by Hearst), the tasks are successively applied to texts, and combined with supervised tasks and techniques
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

4

TOOLS
Outline

1. Introduction
2. Tools for Text Mining
3. IBM Intelligent Miner for Text
4. University of Sheffield General Architecture for Language Engineering (GATE)
5. University of Waikato Environment for Knowledge Analysis (WEKA)

NOTE: all tools are a trade mark of their respective companies

Introduction

• The goals of this track are
  – Briefly reviewing the TM tools market state of the art with special attention to consolidated systems
  – Give an overview of a commercial tool and a research tool
  – Introduce the tool to be used in this tutorial, WEKA
Tools for Text Mining I

- Many KDD tools can be used for TM, provided you add text processing functions
- But the increasing market of specialized commercial TM tools shows the interest of the topic
- A review in [Tan 99] discusses only 11 products, but The Data Warehousing Information Center lists 100 tools and vendors (see the web page)

Tools for Text Mining II

- Most tools provide a subset of the following functionalities
  - Information Extraction
  - Text Retrieval
  - Text Categorization
  - Text Clustering
  - Text Summarization
  - Visualization
  - And a sample of word level techniques (e.g. POS Tagging)
- But nearly none is a “real” TM tool (in Hearst’s sense)
IBM Intelligent Miner for Text I

• A very representative example of state of the art commercial TM environment is IBM Intelligent Miner for Text (IMT)
• Provides support for most of the listed tasks plus some other useful functionalities
• It is sketched in [Tkach 98]

IBM Intelligent Miner for Text II

• IBM IMT provides support for
  – Language recognition
  – Named entity recognition
  – Document clustering (HAC)
  – Text categorization (rule induction, kNN)
  – Text retrieval
  – Text summarization
  – Some support for other languages than English
• And other helpful elements as web spiders
IBM Intelligent Miner for Text III

- IBM IMT has been conceived as TM library, oriented to user design of
  - Business applications, e.g.
    - Customer e-mail processing
    - The journalist’s workstation
  - Intranet/Internet applications, e.g.
    - “Show me more like this” powered searches
    - Topic dependent searches

IBM Intelligent Miner for Text IV

Clustering example (borrowed from IBM IMT Getting Started)
IBM Intelligent Miner for Text V

Feature extraction (borrowed from IBM IMT Getting Started)

Sheffield’s GATE I

- The University of Sheffield General Architecture for Text Engineering (GATE) is architecture, framework and development environment for Language Engineering (LE)
- GATE includes more than 700 Java classes (more than 15 Mb of byte code) and it is open source software (under GNU's GPL)
- See [Cunningham et al. 02] for an overview
Sheffield’s GATE II

- It has been designed to
  - Cleanly separate a number text processing tasks
  - Allow automatic measurement of performance
  - Reduce integration overheads
  - Provide a set of basic language processing components to extended or replaced

- As an architecture
  - It defines the organization of a LE system and the assignment of responsibilities to different components
  - It ensures the component interactions satisfy the system requirements

Sheffield’s GATE III

- As a framework
  - It provides a reusable design for LE systems, and a set of prefabricated software blocks to be used, extended or customised to meet specific needs

- As development environment
  - It helps its users to minimise the time spent to build of modify LE systems, e.g with the debugging mechanism
Sheffield’s GATE IV

• GATE is best understood through an application
• ANNIE (A Nearly-New Information Extraction system) is a IE tool designed to extract relevant information about people from their home pages
• ANNIE components form a pipeline (as text processing is data-intensive, in cascade)
  – See GATE documentation
Sheffield’s GATE VI

- What GATE lacks of
  - Learning machinery!!!
- But this role can be played by other KDD tools, being WEKA a very suitable one

WEKA I

- WEKA is the Waikato Environment for Knowledge Analysis
- It is a fully reusable set of KDD tools including feature selection, ML algorithms, evaluation and visualization
- It consists of around 360 Java classes (more than 1Mb of byte code) and it is open source software (under GNU’s GPL)
- See [Witten & Frank 99] for details
WEKA II

• It is important to note that WEKA is not text oriented
• Currently, you can
  – Either program yourself the text processing tools required
  – Use other packages (e.g. Smart, etc) for text and manage the integration
  – Join and follow up WETA

WEKA III

• WETA (Waikato Environment for Text Analysis) is an initiative framed in the OpenNLP project
• Its goal is to develop a highly scalable solution for text analysis based on machine learning algorithms contained in WEKA
• Still far from it
WEKA IV

• WEKA usage modes
  – For testing different approaches to a learning problem (command-line, GUI)
  – For developing applications that make use of learning
  – For researching in and developing new algorithms

WEKA V

• WEKA for testing learning approaches
  – Most classes provide a main method for command-line usage (designed for testing)
  – A relatively sophisticated GUI is provided, with three options
    • Simple CLI – acts as a command line interface
    • Explorer – designed for processing, learning, evaluation and visualization
    • Experimenter – designed for distributing intensive processing experiments
WEKA VI

• WEKA Explorer
  – Tabs for processing, classifying, clustering, compute associations, perform attribute selection, and visualization
  – A typical operation procedure involves
    • Loading data in Preprocess
    • Visualizing data in Visualize
    • And iteratively
      – Perform attribute selection
      – Test (several) learning algorithm(s)

WEKA VII

Preprocessing tab in WEKA Explorer
WEKA VIII

Classification tab in WEKA Explorer

WEKA IX

Visualizing a threshold curve in WEKA Explorer
WEKA X

Visualizing a decision tree in WEKA Explorer

WEKA XI

- WEKA for developing learning based applications
  - It provides a comprehensive API that allows the development of applications
  - E.g. The tutorial example implements a simple email message recommendation program that learns (kNN) your interests regarding a number of keywords
Summary

- A wide range of “shallow” TM commercial tools, covering many useful tasks and oriented to enterprise environments
- Some research but still useful open source tools
- We believe it is possible to develop high quality, advanced TM applications with the GATE + WEKA combination
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

5

DETECTING PORNOGRAPHY
Outline

1. Motivation
2. Current Technology
3. The POESIA Project
4. POESIA Text Filter by the UEM

Motivation I

- Current social interest in filtering & blocking solutions for a range of environments
- E.g. Kids surfing for fun at the school or library
- E.G. Employees surfing for fun at work
- The common feature is misusing Internet in a place in which it should be used for a different purpose
Motivation II

• There is a market and an industry
  – Internet users most frequent search is sex (by February 2001)
  – The 8.5% of search engine queries deal with sex and pornography (2001)
  – 27.5 million of U.S citizens visited pornographic websites in January 2002
  – U.S. citizens spent $220 million on 2001

Motivation III

• But also there is a problem
• E.g. for kids
  – One of five 10 to 17 years youngsters was asked for sex in 2000
  – One of four had access to unwanted explicit sex stuff
• E.g. for companies
  – Internet abuse at the workplace produced $1 billion loss in 2001
Motivation IV

• On kids safe access to the Internet, there is an ongoing activity by a number of government agencies, including e.g.
  – The Safer Internet Action Plan by the European Commission
  – The ITAS by the US National Research Council
  – NetAlert and the Australian Broadcasting Authority
• On employees Internet misuse at the workplace, there is e.g. an interesting monographic issue in Communications of the ACM (January 02)

Current Technology I

• In recent years, some reports about Internet filtering technology and effectiveness have been published, e.g.
  – NetProtect Report on Filtering Techniques and Approaches
  – CSIRO Report on Effectiveness of Internet Filtering Software Products
• There is an increasing number of commercial and research solutions in the market
Current Technology II

- Current commercial products include some the following techniques
  - Black and white lists
  - Self and third-party labelling (ICRA, PICS)
  - Keyword based text processing
  - Image processing by skin detection
- Most techniques have been found quite ineffective in isolation and are rarely used in combination

Current Technology III

- To our knowledge, there is only one research paper dealing with text based pornography detection (project FILTERIX) [Chandrinos et al. 00]
- They address the problem as TC with
  - Text representation as binary weight vectors
  - Information Gain for feature selection
  - Naive Bayes learning for classification
- With promising effectiveness results
The POESIA Project I

- The work described hereafter is a part of the POESIA project
- POESIA stands for Public Open-source Environment for a Safer Internet Access
- POESIA aims to develop, test, evaluate and promote a fully open-source, extensible, state of the art, filtering and caching software solution, targeted for situations where browsing and other Internet activities are undertaken, e.g. classrooms
- See http://www.poesia-filter.org

The POESIA Project II

- POESIA will
  - Cover at least Web and incoming email channels
  - Filter at least the pornographic and offensive speech domains
  - Target English, Spanish and Italian languages
- Starting on Feb. 2001, and 2 years long
- Partly funded by the EC Safer Internet Action Plan
The POESIA Project III

• Partners at POESIA
  – Istituto di Linguistica Computazionale (Italy)
  – Commissariat à l’Energie Atomique (France)
  – Ecole Nouvelle d’Ingénieurs en Communication (France)
  – M.E.T.A. S.r.l. (Italy)
  – Universidad Europea de Madrid CEES (Spain)
  – University of Sheffield (UK)
  – Fundació Catalana per a la Recerca (Spain)
  – PiXEL Associazione (Italy)
  – Liverpool Hope University College (UK)
  – Telefónica Investigación y Desarrollo (Spain)

The POESIA Project IV

• Technologic approach
  – Combination of a number of techniques including
    • Label detection
    • Sophisticated text analysis
    • Sophisticated image processing
    • Script code analysis
  • Effectiveness is got by applying state-of-the-art research approaches
  • Efficiency is got through a two level schema and caching facilities
The POESIA Project V

• We focus on the text processing approaches
  – Under development by Sheffield, ILC and UEM
  – Two-stage architecture
    1. A simple (‘lite’) filtering agent which makes only light use of NLP techniques, and can rapidly process large text volumes
    2. A sophisticated (‘heavy’) filtering agent which makes heavier use of NLP resources and techniques to filter only those documents that are left uncategorized by the first agent
  – Addressed as a TC task

The POESIA Project VI

• The most sensible approach for the lite text filter, given current state of the art, is
  – Text representation based on binary or $tf.idf$ weight vectors
  – Feature selection with Information Gain, $\chi^2$, etc
  – Learning with Support Vector Machines and possibly some cost sensitive method (e.g. MetaCost)
The POESIA Project VII

• Current Spanish text filtering prototype
  – Techniques
    • Binary weight vectors
    • IG
    • SVM
  – Effectiveness
    • English accuracy 92%
    • Spanish accuracy 88%

The POESIA Project VIII

• Current Spanish text filtering prototype
  – Technology
    • 26 Java classes
    • 2600 code lines
    • Reusing
      – WEKA
      – HTMLParser
      – Muffin proxy filter demonstration
The POESIA Project IX

- Current Spanish text filtering prototype
  - Design
    - A HTML parsing package
    - A binary index package
    - Two Muffin filter classes
  - Operation
    - Learning step
      - A set of manually classified web pages are indexed and a WEKA SVM classifier trained on them
    - Classification step
      - The demanded web page is processed and classified on-the-fly
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

6

DETECTING SPAM
Outline

1. Motivation
2. State of the Art
3. Problem Description
4. Evaluation Framework
5. Comparison of Approaches
6. Results and Conclusions
7. Notes about implementation

Motivation

• Spam email is more properly, Unsolicited Bulk Email (UBE)
• It has been producing a considerable damage to
  – Internet Service Providers
  – Internet users
  – and the whole Internet backbone
• For instance, Internet subscribers worldwide are wasting an estimated 10 billion euro a year just in connection costs due to UBE
State of the Art I

- Three kinds of proposals to address UBE [Cranor & LaMacchia 98, Hoffman & Crocker 98]
  - Economic
    - Charging sending email
  - Regularory
    - Law definition and enforcement
  - Technical
    - Filtering mechanisms

State of the Art II

- Technical approaches
  - Channels [Hall 99]
  - Aliasing [Gabber et al. 99]
  - Filtering
    - Black & white lists
    - Bulk message detection
    - Content-based detection
      - Manual filters (e.g. the one operated by BrightMail)
      - Machine learned classifiers (see e.g. [Sahami et al. 98, Gómez et al. 00, Gómez et al. 02] and the bibliography at http://linwww.ira.uka.de/bibliography/Ai/MLSpamBibliography.html)
State of the Art III

• Very good reported results for manual filters in a study by ETesting Labs [Etesting Labs 01]

• Brightmail approach seems very effective
  – Able to catch 93.9% of UBE without misclassifying any legitimate message
  – Based on millions of email addresses receiving UBE and a team of experts manually codifying rules on-the-fly

Problem Description I

• UBE detection is a TC problem
  – Two classes (UBE and legitimate email)
  – It relatively easy to
    • Represent messages as vectors of concept weights
    • Perform some feature selection
    • Learn a classifier

• But evaluation is not so simple because it is a problem in which misclassification costs and class distribution are not symmetric
Problem Description II

• It is clear that users prefer dealing with more UBE to missing legitimate email
• But the preference is not rated anywhere, i.e. we do not know the relative costs of both kinds of mistakes
  – E.g. It is one hundred times worse missing a legitimate email than receiving a UBE?
• Even worse, these costs may vary

Problem Description III

• So we need
  1. A method for evaluating classification accuracy that is independent of class and cost distributions
  2. To consider cost-sensitive learning methods (e.g. stratification and weighting, threshold variation, MetaCost, BoostCost, etc)
Evaluation Framework I

- We use the Receiver Operating Characteristic Convex Hull (ROCCH) method [Drummond & Holte 00, Provost & Fawcett 97, 01]
- ROC analysis allows a visual comparison of the performance of a set of ML algorithms, regardless of the class and cost conditions
- With the study of the Convex Hull, we can detect the best approach for the required class and cost distributions

Evaluation Framework II

- A ROC curve for an algorithm is produced by
  - Learning classifiers for a variety of conditions and linking the set of (false positive, true positive) points in a 2D graph
  - Or going through the rank of classified instances and obtaining the set of (fp,tp) points
- We follow the first approach
  - It is rather time consuming but you can store the learned classifiers and use the best for some application environment
Evaluation Framework III

- Sketch of evaluation method
  1. For each ML algorithm, obtain a ROC curve and plot it (or only its convex hull) on the ROC space
  2. Find the convex hull of the set of ROC curves previously plotted
  3. Find the range of slopes for which each ROC curve lies on the convex hull
  4. In case that target conditions are known, compute the corresponding slope value and output the best algorithm. In other case, output all ranges and best local algorithms or classifiers

Evaluation Framework IV

- Given a class and cost distribution a slope can be computed as
  \[ m = \frac{c(Y,n) \cdot P(n)}{c(N,p) \cdot P(p)} \]
- Being c(Y,n) and c(N,y) the cost of a false positive and a false negative
- And P(y) and P(n) the probabilities of positive and negative classes
Comparison of Approaches I

• We have compared a number of TC approaches for detecting UBE (see [Gómez 02]) with
  – Text representation as binary and tf.idf weight vectors
  – Feature selection with IG, reducing the original concept space to 1%

Comparison of Approaches II

• We have tested a number of ML algorithms
  – C4.5
  – Naive Bayes
  – The rule learner PART
  – Support Vector Machines
  – The Rocchio algorithm
Comparison of Approaches III

• We have tested a number of ML methods for making algorithms cost-sensitive
  – The Threshold method
  – The Weighting method (equivalency to stratification by oversampling)
  – The MetaCost method

Results and Conclusions I

ROCCH curve for the comparison of best classifiers and cost-sensitive methods
Results and Conclusions II

<table>
<thead>
<tr>
<th>Slope Range</th>
<th>(FP, TP) point</th>
<th>Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0.000, 0.010]</td>
<td>(0.206, 1.000)</td>
<td>PAMc040</td>
</tr>
<tr>
<td>[0.010, 0.044]</td>
<td>(0.108, 0.999)</td>
<td>SVWE005</td>
</tr>
<tr>
<td>[0.044, 0.357]</td>
<td>(0.040, 0.996)</td>
<td>SVTH001</td>
</tr>
<tr>
<td>[0.357, 1.250]</td>
<td>(0.012, 0.986)</td>
<td>ROTH020</td>
</tr>
<tr>
<td>[1.250, 14.750]</td>
<td>(0.004, 0.976)</td>
<td>NBWE600</td>
</tr>
<tr>
<td>[14.750, \infty]</td>
<td>(0.000, 0.917)</td>
<td>SVWE200</td>
</tr>
</tbody>
</table>

Optimality ranges for the best classifiers and cost-sensitive methods

Results and Conclusions III

- Regarding cost-sensitive methods
  - No one is clearly superior
  - Instance Weighting is the most frequent winner
- Regarding learning algorithms
  - No one is clearly superior
  - SVM is the most frequent winner
Results and Conclusions IV

• There are three scenarios considered important in the literature, corresponding to cost distributions in which a false positive is 1, 9 and 999 worse than a false negative

<table>
<thead>
<tr>
<th>Cost Ratio</th>
<th>Slope</th>
<th>Best Classifier</th>
<th>R</th>
<th>P</th>
<th>WA</th>
<th>TCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.014</td>
<td>NBWE600</td>
<td>0.976</td>
<td>0.979</td>
<td>0.992</td>
<td>22.697</td>
</tr>
<tr>
<td>9</td>
<td>45.13</td>
<td>SVWE200</td>
<td>0.917</td>
<td>1.000</td>
<td>0.999</td>
<td>12.048</td>
</tr>
<tr>
<td>999</td>
<td>5009.538</td>
<td>SVWE200</td>
<td>0.917</td>
<td>1.000</td>
<td>0.999</td>
<td>12.048</td>
</tr>
</tbody>
</table>

Out results for the scenarios

Results and Conclusions V

• Our results are better than those reported by others
• More interestingly, for extreme conditions, we are close to real world manual performance
  – Brightmail detects 93.1% UBE without false positives
  – We get 91.7% UBE without false positives
Notes about Implementation

- We used a Ling-spam test collection
- We processed the messages with the retrieval engine Smart to get the representation
- But the bulk of the work has been done with WEKA
- With our work in POESIA, we can use the implemented indexing tools to allow interacting with Smart
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

7

CHALLENGES
The challenge: supporting real TM

• There are a number of TM related tools
  – They provide support for advanced text analysis tasks
  – They allow to discover new knowledge to the user but not the author
• But we defined real TM as discovering absolutely new knowledge

The Challenge: supporting real TM

• E.g. The example cited by Heast [99], describing Swanson’s work
  – Given
    • medical titles and abstracts
    • a problem (incurable rare disease)
    • some medical expertise
  – find causal links among titles
    • symptoms
    • drugs
    • results
The Challenge: supporting real TM

• E.g. The example cited by Heast [99]
  – Problem: Migraine headaches (M)
    • stress associated with M
    • stress leads to loss of magnesium
    • calcium channel blockers prevent some M
    • magnesium is a natural calcium channel blocker
    • spreading cortical depression (SCD) implicated in M
    • high levels of magnesium inhibit SCD
    • M patients have high platelet aggregability
    • magnesium can suppress platelet aggregability
  – All extracted from medical journal titles

The Challenge: supporting real TM

• E.g. The example cited by Heast [99]
  – These clues suggest that magnesium deficiency may play a role in some kinds of migraine headache
  – The hypothesis which did not exist in the literature at the time Swanson found these links
  – The hypothesis must of course be confirmed experimentally
  – The process to derive the hypothesis was not automatic
The Challenge: supporting real TM

• So what we need is tools to strongly support this kind of knowledge discovery
• Hearst describes a project called LINDI focused on developing this kind of tool for finding functions of genes
• The idea behind the system is mixed-initiative interaction
  – User applies tools to help explore the hypothesis space
  – System runs suites of algorithms to help explore the space, suggest directions

The Challenge: supporting real TM

• The system has three main parts
  – UI for building/using strategies
  – Backend for interfacing with various databases and translating different formats
  – Content analysis/machine learning for figuring out good hypotheses/throwing out bad ones
The Challenge: supporting real TM

- A point is that we have open-source tools to develop this kind of systems, in a three layered architecture
  - GATE may act as backend
  - WEKA+GATE may act the learning-based middleware
  - UI and integration are required

- But still we need further investigation in several fields [Tan 99], specially
  - Multilinguality
  - Domain knowledge integration

- And of course all the challenges in KDD
Text Mining and Internet Content Filtering

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/

REFERENCES
Text Mining and Internet Content Filtering
References for the ECML/PKDD Tutorial, August 19th, 2002

José María Gómez Hidalgo
Departamento de Inteligencia Artificial
Universidad Europea CEES
jmgomez@dinar.esi.uem.es
http://www.esi.uem.es/~jmgomez/


[Cranor & LaMacchia 98] Lorrie F. Cranor and Brian A. LaMacchia. Spam! Comm. of the ACM, 41(8), 1998.


