Why automated classification?

- Information explosion
  - Documents increasingly available electronically
  - Lots of unstructured full-text documents on the Web
- High cost of manual classification (1-2 / hour)
- Challenging research issue
- Fun!

Outline

- Background
- SVM (Support Vector Machines)
- String matching
- Evaluation
- Lessons learned
- References

History ...

(Personal - NetLab)

1992  Automated classification of WAIS databases using 2 top levels of UDC
1993  Demonstrated at SIGWAIS/SIGNIDR III conference
1997  Automated classification of Engineering Web resources using Ei
2000  EU project DESIRE: toolkit (Matcher)
2003  EU project ALVIS: Matcher + Crawler => Focused Web crawler (Combine)
2007  PhD thesis: “Automated Subject Classification of Textual Documents in the Context of Web-based Hierarchical Browsing”, Koraljka Golub
2009  Vertical Search Engines Demo
Automated Classification technologies

- Machine learning methods
  - Statistical models (Bayes, SVM, ...)
  - ANN
- Information Retrieval methods
  - Clustering (no predefined categories)
- Library Science methods
  - String matching + Thesaurus

Outline

- Background
- SVM (Support Vector Machines)
- String matching
- Evaluation
- Lessons learned
- References

### SVM

- Developed by Vapnik 1992
- Classification for linear (and non-linear) problems
  - "Kernels" handle non-linear problems (by mapping to linear case)
- Machine learning
- Data represented as n-dimensional vectors (vector space model)
- Need a training set with positive and negative documents
- General classifier
- Decision: yes/no
- Finds the optimal hyper-plane for linearly separable patterns
- Can be extended to multiclass/hierarchical classification
- Efficiently handles $\approx 10,000$ dimensions given that input vectors are sparse
- Decision function specified by support vectors (from training examples)
- SVM maximize the margin around the separating hyper-plane
Why SVM for text categorization?

**Advantages**
- "Most popular and effective method"
- High dimensionality input
- Uses all features - no feature selection
- Sound mathematical theory for optimal decision function
- Performs well when collection characteristics does not change
- Bag-Of-Words model - document vectors
- Fast once trained

**Problems**
- Requires training examples
- Language
- Depends on a relatively homogeneous collection
- Sensitive for selection of negative examples
- Error propagation for deep classification hierarchies
- One classifier per class

Outline
- Background
- SVM (Support Vector Machines)
- String matching
- Evaluation
- Lessons learned
- References

Classification process

Document text

**Configuration**
- Stop-words
- Stemming

**String Match**

**Score propagation**
- Cut-off values

**Term Definition**
- Term triplets
- Term (word, phrase boolean), relevance, list of topic-classes

**Topic-class hierarchy**

List of topic-classes, relevance, matched terms

Example term triplets
- 40: ALGOL @and programming languages=723.1.1
- 15: CCTV=716.4
- 40: CAT scans=723.5
- 20: CAT scans=531, 801, 461.1
- -10000: hotel=7

String matching

**Thesauri based**
- Reuse intellectual effort
- Topic terms (features) from thesaurus
  - ... are they present in the text?
  - ... relevance: how many; where in the text (document structure)

Relevance score =

\[
\sum_{\text{all locations}} \left( \sum_{\text{all terms}} \text{hits}[\text{location}][\text{term}] \times \text{weight}[\text{term}] \times \text{weight}[\text{location}] \right)
\]

or

\[
\sum_{\text{all terms}} \left( \sum_{\text{all matches}} \log(k \times \text{position}[\text{term}][\text{match}] + \text{proximity}[\text{term}][\text{match}]) \right)
\]

Normalize with respect to document size
Why String matching for text categorization?

Advantages
- Reuse intellectual effort
- Can take advantage of document structure
- Feature selection by thesaurus
- Language
- No training
- Deep hierarchies
- Multiclass classification

Problems
- No context for topic terms
- Stopwords can cause trouble
- Relies on a good thesaurus
- No generalization

Outline
- Background
- SVM (Support Vector Machines)
- String matching
- Evaluation
- Lessons learned
- References

Evaluation challenge
Comparing human assigned classes to automated classification
- Collection policies
- Users vs indexers
- Inter- and intra-indexers consistency
- Availability of representative pre-classified collections

Hard to do good evaluations

Evaluation
- SVM
  - Most evaluations done in "lab-like environments"
  - Very good - 70 - 90 % correctness
  - Popular
- String matching
  - Few evaluations done
  - Good - 60 - 90 % correctness

Examples:
1: Precision for classification of Compendex bibliographic records:
   SVM 0.74 - 0.91
   String match 0.26 - 0.97
2: Depends on the hierarchical depth of the classification
   Correct to String match SVM
   3 levels 0.71p 0.61p
   2 levels 0.87p 0.81p
   top level p p
Outline

1. Background
2. SVM (Support Vector Machines)
3. String matching
4. Evaluation
5. Lessons learned
6. References

Lessons learned I

- Homogeneous collection
- Good training examples (both positive and negative)
- Shallow hierarchy
- Mixed collection
- Good thesaurus with subject terms
- Multiple classes in a hierarchy

Lessons learned II

- Careful with text preprocessing (stopwords and stemming)
- Hard to do a good evaluation
- Learn strengths and weaknesses
- Experiment!
- There is no “fit all cases best” solution
- Not perfect
  - ... but useful

Idea - compromise and use all

- Start with a reasonable classification system and thesaurus
- Collect documents and classify by String Matching
- Use result to generate SVM training set
- Train SVM classifier
- Reclassify all documents
- Manually inspect results and update SVM training set
- Go-to 3 until result good enough
- Production level service
Outline

1. Background
2. SVM (Support Vector Machines)
3. String matching
4. Evaluation
5. Lessons learned
6. References

References/More info

- This presentation: [http://combine.it.lth.se/UDCseminar2009/](http://combine.it.lth.se/UDCseminar2009/)
- Koraljka Golub PhD thesis: “Automated Subject Classification of Textual Documents in the Context of Web-based Hierarchical Browsing”
- Combine focused crawler tools download: [http://combine.it.lth.se/#downloads](http://combine.it.lth.se/#downloads)
- Documentation on automated classification: [http://combine.it.lth.se/documentation/DocMain/node6.html](http://combine.it.lth.se/documentation/DocMain/node6.html)
- Demonstrators (incl UDC classifiers): [http://dbkit05.eit.lth.se/exp/Demos/](http://dbkit05.eit.lth.se/exp/Demos/)