Learning in Information Agents

Tom Mitchell

School of Computer Science
Carnegie Mellon University

January 1997

Opportunities

There is an information flood

- Learn user interests
- Learn to classify documents
- Learn where to look for what
- Learn to extract information from text

Some Examples

Learn user interests

- NewsWeeder → WiseWire
- Homer → Firefly
- Syskill & Webert

Web browsers that learn

- WebWatcher
- Letizia

Outline

- Classifying text documents
  - Bags of words
  - TFIDF
  - Compression
  - naive Bayes
  - classifying speech documents
  - bootstrapping the learner
- Social learning
  - Combining opinions
  - Merging content and social learning
- Learning over hypertext
  - Web browsers that learn
  - Mining the web
- Research challenges

NewsWeeder

[Lang 1995]

Nobody can read 8000+ newsgroups

Many uninteresting articles

The few interesting ones get lost in the flood

Approach:

- Build a newsreader that learns your preferences
- Let it fetch the top articles from all unread newsgroups

How Can We Learn Over Text?

Given:

- 20 different newsgroups
- 1000 training articles from each

Learn:

- To classify which newsgroup a new article is from

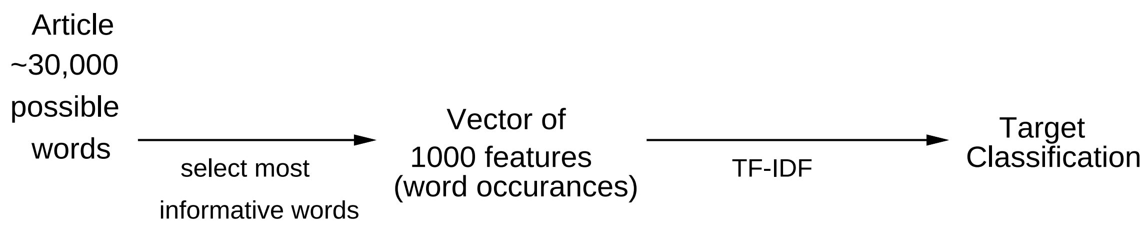
Twenty NewsGroups

comp.graphics	misc.forsale
comp.os.ms-windows.misc	rec.autos
comp.sys.ibm.pc.hardware	rec.motorcycles
comp.sys.mac.hardware	rec.sport.baseball
comp.windows.x	rec.sport.hockey
alt.atheism	sci.space
soc.religion.christian	sci.crypt
talk.religion.misc	sci.electronics
talk.politics.mideast	sci.med
talk.politics.misc	
talk.politics.guns	

Learning Over Arbitrary Text Items

Key idea: represent text document as a feature vector

One feature per vocabulary word



TFIDF

[Salton]

Represent document by vector of word weights

Weight for ith word in document d from set D

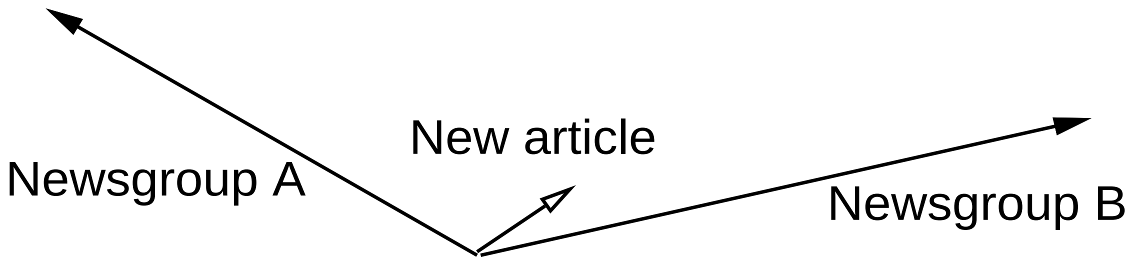
$$w(i, d) \equiv tf(i, d) \cdot idf(i)$$

where

- $tf(i, d) = \#$ of times word i occurs in d
- $idf(i) = \log\left(\frac{|D|}{df(i)}\right)$
- $df(i) = \#$ of documents from D that contain word i

Several variants, developed for information retrieval

Using TFIDF to classify documents



Accuracy for 20 newsgroups task: 82%

Unix Compress Method

[Lang, Caruana]

Train:

- For each $class_j$ of documents
  - create file f_j containing all documents from $class_j$
  - run Unix compress on f_j

Classify:

- To classify new document *doc*:
  - add *doc* to each file f_j and recompress
  - assign *doc* to class that best compresses it

$$class(doc) \leftarrow \operatorname*{argmin}_j compress(f_j + doc) - compress(f_j)$$

Bayesian Method

What would optimal* compression produce?

Information theory: $-\log_2 p$ bits is optimal

Train:

- For each $class_j$ of documents

  For each word w_i

    estimate $P(w_i|class_j)$

Classify:

- assign *doc* to class that best compresses it

$$class(doc) \leftarrow \underset{j}{\operatorname{argmin}} \sum_{w_i \in doc} -\log_2 P(w_i|class_j)$$

* assuming words independent (Naive Bayes)

Naive Bayes: Details

LEARN_NAIVE_BAYES_TEXT($Examples, V$)

- $Vocabulary \leftarrow$ the set of all distinct words
- For each target value v_j in V do
  - $docs_j \leftarrow Examples$ that have target value v_j
  - $P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$
  - $Text_j \leftarrow$ concatenate all members of $docs_j$
  - $n \leftarrow$ number of word positions in $Text_j$
  - for each word w_k in $Vocabulary$
    - $n_k \leftarrow$ number of times word w_k occurs in $Text_j$
    - $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$

Applying Naive Bayes Classifier

CLASSIFY_NAIVE_BAYES_TEXT(Doc)

Return the estimated target value for the document Doc. a_i denotes the word found in the ith position within Doc.

- $positions \leftarrow$ all word positions in Doc that contain tokens found in $Vocabulary$
- Return v_{NB}, where

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_{i \in positions} P(a_i|v_j)$$

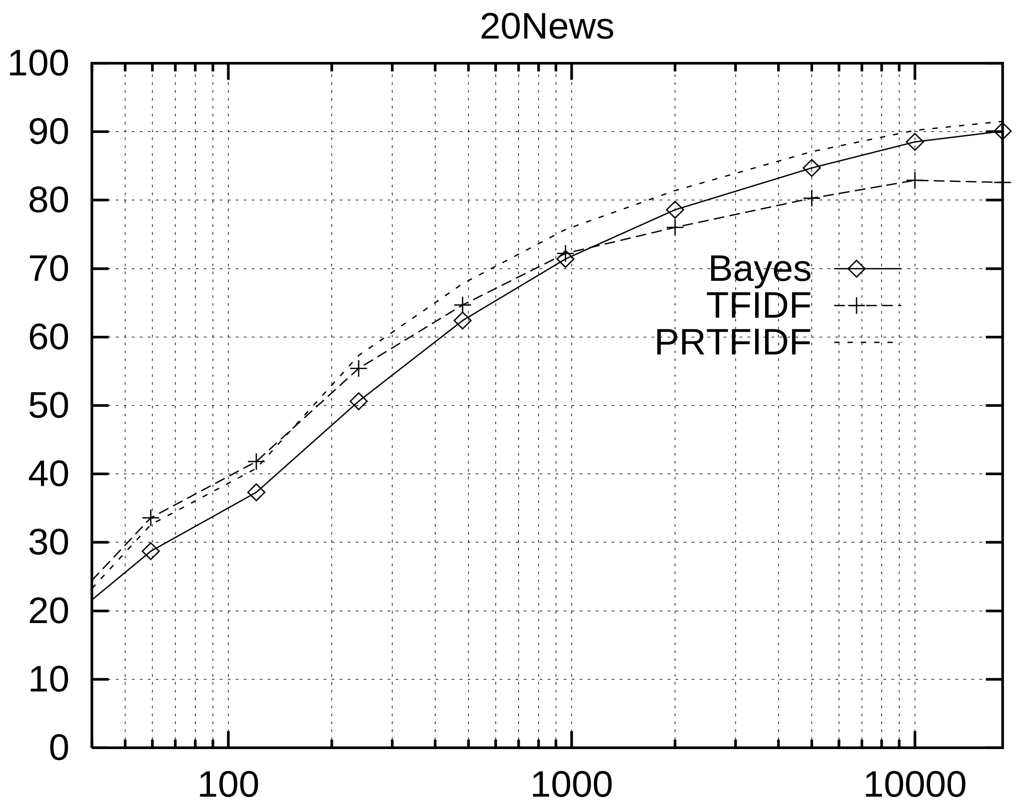
Naive Bayes: Subtleties

- Note sum over words in *document* rather than words in *vocabulary*
- Use m-estimate to avoid problems with rare words

$$P(w_k|v_j) \leftarrow \frac{n_k + 1}{n + |Vocabulary|}$$

- To calculate
  $P(\text{word } w_k \text{ found in position } a_i \text{ of document } d|v_j)$
  - Assume cond. indep. of other words in d, given v_j
  - Assume cond. indep. of position a_i, given v_j
    $$P(a_i = w_k|v_j) = P(a_r = w_k|v_j), (\forall i, r)$$

Learning Curve



Accuracy vs. Training set size (1/3 withheld for test)

Results of Learning Newsgroup Identity

- Expected/Default = 5% correct
- Unix compression Learning = 52%
- TFIDF = 82%
- naive Bayes = 89%
- Others = 41% - 63%

Does NewsWeeder Retrieve Interesting Articles?

Percent of articles user rates interesting (1 or 2)

	UserA	UserB
All articles	11%	20 %
NewsWeeder selections	51%	65 %

"NewsWeeder selections" are its top 10%

Other Learning Information Filters

- SYSKILL & WEBERT for web pages, Pazzani et al. (1996)
- RINGO, HOMER → FIREFLY for music, movies, Shardanand and Maes (1995).

Other Approaches to Text Classification

- Rule and decision tree learning (e.g., Cohen (1995), Apte et al. (1994))
- Linear classifiers (e.g., Yang and Chute (1994), Lewis et al. (1996))
- Clustering-based methods (e.g., Hearst and Pedersen (1996), Smyth (1996), Sahami et al. (1996))
- Simulated evolution methods (e.g., Maes (1994), Moukas (1997))

Best approach seems to depend on features of the data, such as

- Mean document length
- Variance in document length
- Class distribution

Social Learning

[Shardanand and Maes 1995]

Idea: Learn by correlating user ratings

	d_1	d_2	d_3	d_4	d_5	d_6
User A	2	-	4	4	-	5
User B	1	5	4	-	3	4
User C	5	2	-	2	1	-
User D	3	-	2	2	-	4

Predict

$$rating(user_i, d_n) = \frac{\Sigma_{i \neq j}\, w_{ij} \cdot rating(user_j, d_n)}{\Sigma_{i \neq j}\, w_{ij}}$$

where w_{ij} is a measure of correlation of $user_i$ and $user_j$ interests

Social Learning, Cont'd.

Base algorithm: (i.e., mean)

$$w_{ij} = \frac{1}{\text{number of users}}$$

Mean sq difference:

$$w_{ij} = \frac{1}{\Sigma_d (r(U_i, d) - r(U_j, d))^2}$$

Pearson r correlation:

$$w_{ij} = \frac{\Sigma_d (r(U_i, d) - \overline{U_i})(r(U_j, d) - \overline{U_j})}{\sqrt{\Sigma_d (r(U_i, d) - \overline{U_i})^2 (r(U_j, d) - \overline{U_j})^2}}$$

Constrained *Pearson r* correlation (4 = neutral):

$$w_{ij} = \frac{\Sigma_d (r(U_i, d) - 4)(r(U_j, d) - 4)}{\sqrt{\Sigma_d (r(U_i, d) - 4)^2 (r(U_j, d) - 4)^2}}$$

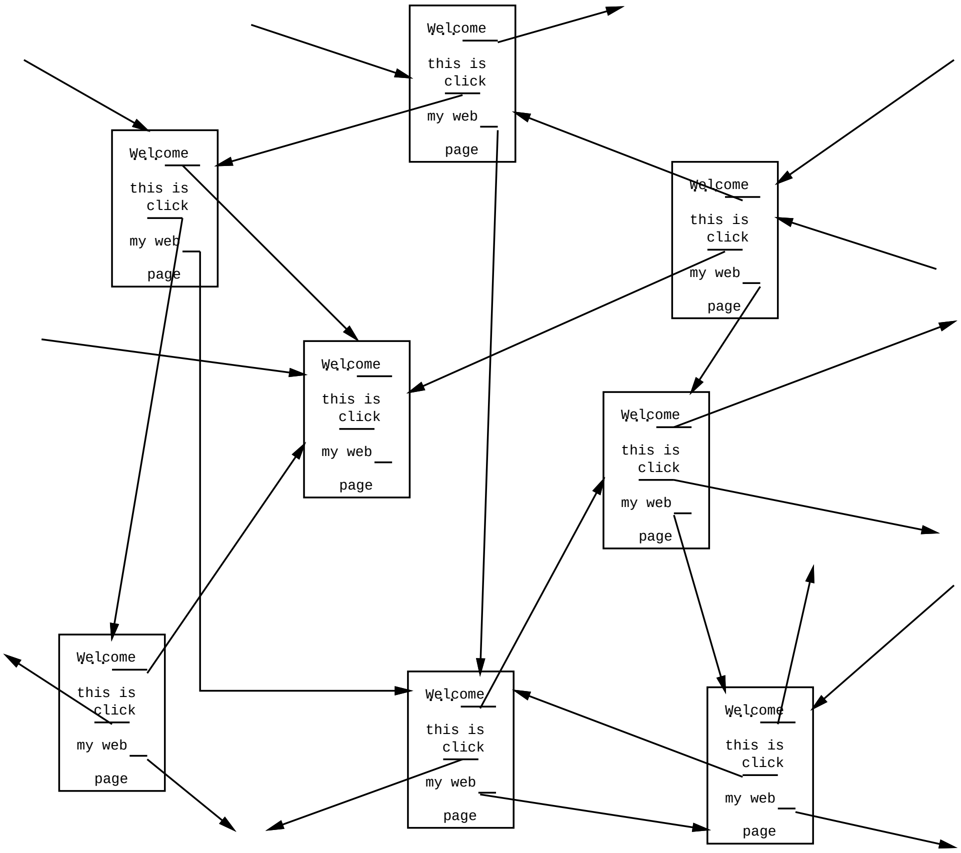
Results of Social Learning in Ringo

2000 users, 125 musical artists, ratings 1–7

Thresholded sum, considering only terms where $w_{ij} > threshold$

- Base alg: mean error = 1.3, $\sigma = 1.6$
- Constrained Peason r: mean error = 1.1, $\sigma = 1.3$, 97% of cases predictable

The World Wide Web



WebWatcher learning task:

$$User \times Goal \times Page \times Trajectory \times H{-}link \rightarrow \{1, 0\}$$

WebWatcher Learning Methods

Build word vector by

1. collecting words "near" the hyperlink
2. accumulating goals of users traversing this hyperlink
3. collecting discounted future words

WebWatcher Learning Performance

Evaluating learning methods:

1. Suggest up to 3 hyperlinks on current page
2. Measure how often user followed suggested link

Method	Accuracy
Random guessing	~22%
Human experts*	47%
Hyperlink/Nearby Words	39%
Accumulate Users' Goals	41%
Discounted Future Words	45%
Combine Above Methods	49%

* humans tested only on SCS Front door page

World Wide Knowledge Base

The Internet has huge ammounts of useful knowledge

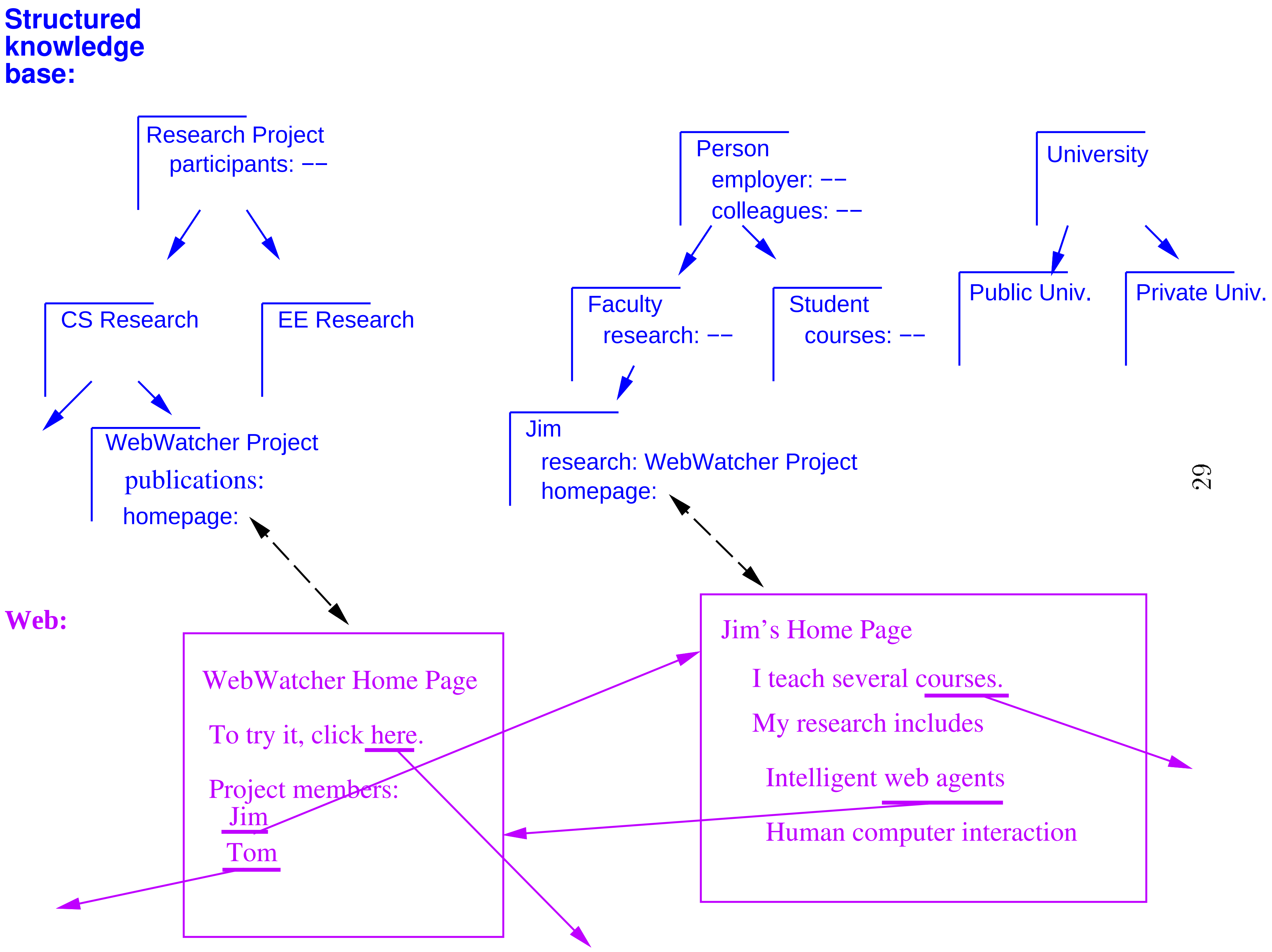
- WWWeb: mostly static
- Online newsfeeds: dynamic

Build trainable system to convert this to symbolic, probabilistic knowledge base

- Trainable for user's choice of ontology
- After training, automatically builds/maintains kb

Why a WWKB?

- More sophisticated queries of web
  - Find Mark's new publications since AAAI96.
- Knowledge-based search strategies for info
  - strategies to find courses on ML
- Knowledge-based problem solving
  - KB system for predicting companies about to merge


Structured knowledge base:
Research Project
participants: --
CS Research
EE Research
WebWatcher Project
publications:
homepage:
Person
employer: --
colleagues: --
Faculty
research: --
Student
courses: --
University
Public Univ.
Private Univ.
Jim
research: WebWatcher Project
homepage:
Web:
WebWatcher Home Page
To try it, click here.
Project members:
Jim
Tom
Jim's Home Page
I teach several courses.
My research includes
Intelligent web agents
Human computer interaction

Approach

Inputs:

- Ontology to define relevant classes, relations
- Training examples of classes, relations

Goal:

- System learns to discover new instances of classes
- and new instances of relations
- then explores the WWWeb to create a KB that mirrors its content

Key research questions:

- How well can system learn to identify new instances, relations?
- Can the growing KB effectively guide learning?

Experiment 1: Classify Web Pages

[Choon Quek]

1546 web pages, mostly from CMU

Predicted category	True category			
	Department	Research	Person	Random
Department	73	0	0	2.7
Research	0	40.4	0	2.1
Person	13	37	97.7	5
Random	14	22	2.3	90

Experiment 2: Classify Hyperlinks

1366 hyperlinks from home personal pages

Predicted category	True category	
	Person's research	Random
Research	110	47
Random	11	286

A Learned Text Extraction Rule

[Freitag, Mitchell]

A rule for location

```
If a word is parsed with a left J-link;
   and it is J-linked to ``in'';
   and it is not capitalized;
   and ``in'' has a left MV-link
Then it is the main location word.
```

A sentence it matches

A talk will follow in Hammerschlag Hall $\boxed{1112}$.

Parser output

```
 +---------------------Xp-------------------------+
 +-Wd---+                  +--------Js------+  |
 | +-Ds-+--Ss--+--I--+-MVp--+    +--G---+--G-+  |
 | |    |      |     |      |    |      |    |  |
// A talk.n will.v follow.v in Hschlg Hall 1112 .
```

References

For more information...

- R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell (1995). Webwatcher: A learning apprentice for the world wide web. *AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, March 1995.
- M. Balabanovic and Y. Shoham. (1995). Learning information retrieval agents: Experiments with automated web browsing, *AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments, Working Notes.* AAAI-Press, 1995.
- Croft, B. and Turtle, H. (1992). Text retrieval and inference, in Jacobs (ed.) *Text-Based Intelligent Systems* Lawrence Erlbaum, 127–156.
- Etzioni, O. (1996). Moving up the information food chain: deploying softbots on the world wide web, *Proceedings of the National Conference on AI*, Portland, OR, August 1996.
- Hearst, M. and Pedersen, J. (1996). Reexamining the cluster hypothesis: Scatee/Gather on retrieval results, *SIGIR96*.
- Joachims, T. (1996). *A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization*, Computer Science Technical Report CMU-CS-96-118, Carnegie Mellon University.
- Lang, K. (1995). Newsweeder: learning to filter netnews, *Proceedings of the 12th International Conference on Machine Learning*, Prieditis and Russell (eds.), Morgan Kaufmann Publishers, San Francisco, pp. 331-339.
- Lashkari, Y., Metral, M., and Maes, P. (1994). Collaborative interface agents. *Proceedings of AAAI '94 Conference*, Seattle, Washington, August 1994.
- Lewis, D. (1991). Representation and learning in information retrieval, Ph.D. thesis, Dept. of Computer and Information Science, University of Massachusetts, COINS Technical Report 91-93.
- H. Lieberman (1995). Letizia: An agent that assists web browsing. *International Joint Conference on Artificial Intelligence, Montreal*, August 1995.

- Maes, P. (1994). Agents that reduce work and information overload, *Communications of the ACM* 37(7) 31–40.
- T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistent. *Communications of the ACM*, 37(7):81–91, July 1994.
- Moukas, A. (1997). Amalthaea: information discovery and filtering using a multiagent evolving ecosystem. To appear *International Journal of Applied Artificial Intelligence.*
- M. Pazzani, J. Muramatsu, and D. Billsus (1996). Syskill & webert: Identifying interesting web sites. *AAAI Conference, Portland*, August 1996.
- Rocchio, J. (1971). Relevance feedback in information retrieval, in *The SMART retrieval system: experiments in automatic document processing*, Ch. 14, Prentice-Hall, pp. 313-323.
- Sahami, M. Hearst, M. and Saund, R. (1996). Applying the mulitple cause mixture model to text catergorization, *Proceedings of the International Conference on Machine Learning*, 435–443.
- Salton, G. (1991). Developments in automatic text retrieval, *Science*, Vol 253, pp. 974-979.
- Shardanand, U. and Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth,", *CHI-95*.