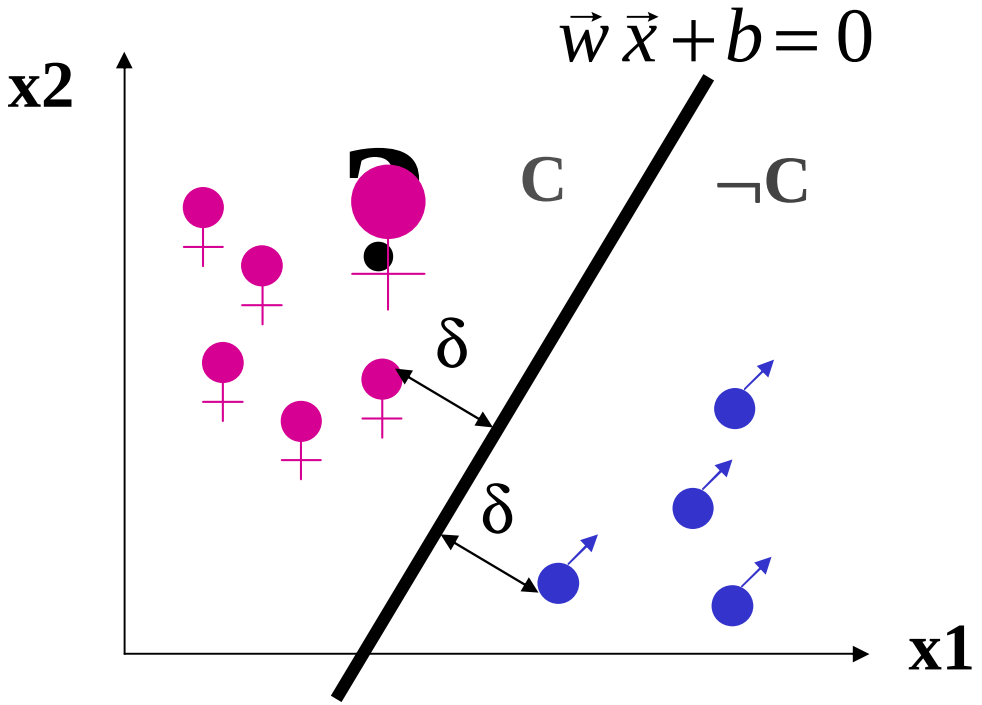
Chapter 6: Automatic Classification (Supervised Data Organization)

6.5 Discriminative Classifiers: Support Vector Machines (SVM) for Binary Classification



n training vectors
$(x_1, ..., x_m, C)$
with C = +1 or -1

large-margin separating hyperplane minimizes risk of classification error

Determine ***hyperplane*** $\vec{w}\,\vec{x}+b=0$ that optimally ***separates*** the training vectors in C from those not in C, such that the (Euclidean) distance δ of the (positive and negative) training samples closest to the hyperplane is maximized. (Vectors with distance δ are called ***support vectors***.)

Classify new test vector $\vec{y}$ into C if: $(\vec{w}\,\vec{y}+b)=\sum_{i=1}^{m} w_i y_i + b > 0$

Computation of the Optimal Hyperplane

Find $\vec{w} \in R^m$ and b $\in$ R such that

1. $\delta \in$ R is maximal and
2. $C_i \frac{1}{\|\vec{w}\|}(\vec{w}\,\vec{x}_i + b) \geq \delta$ for all i=1, ..., n

This is (w.l.o.g. with the choice $\|\vec{w}\| = 1/\delta$) equivalent to (V. Vapnik: Statistical Learning Theory, 1998):

Find $\alpha_1, ..., \alpha_n \in R_0^+$ such that

1. $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} C_i C_j \alpha_i \alpha_j (\vec{x}_i \vec{x}_j)$ is minimal (Quadratic programming problem)
2. and $\sum_{i=1}^{n} C_i \alpha_i = 0$

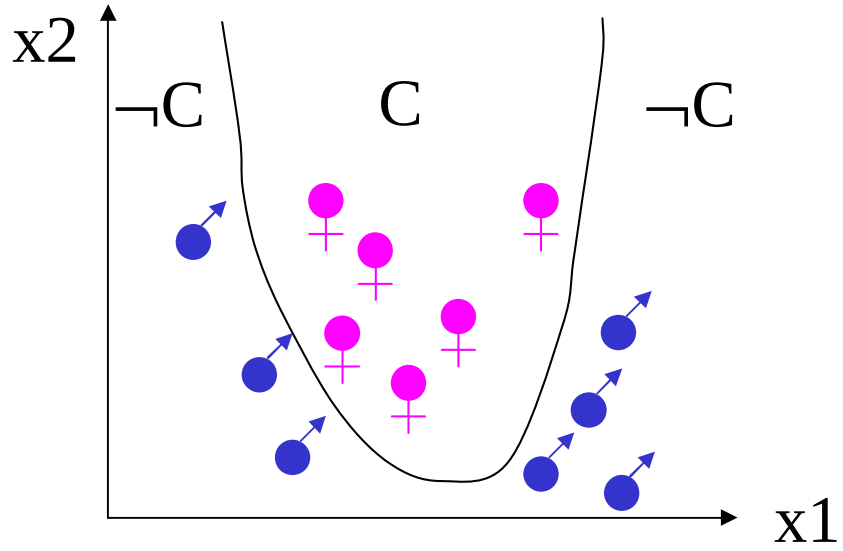
Optimal vector $\vec{w}$ ist linear combination (where $\alpha_i > 0$ only for support vectors)

$$\vec{w} = \sum_{i=1}^{n} \alpha_i C_i \vec{x}_i$$

b is derived from any support vector $\vec{x}_j$ by:

$$b = C_j - \vec{w}\,\vec{x}_j$$

SVMs with Nonlinear Separation



Transform vectors $\vec{x} \in R^m$ into $\Phi(\vec{x}) \in R^{m'}$ with m‘ > m

e.g.: $\Phi((x_1, x_2)) = (ax_1^2, bx_2^2, cx_1x_2, dx_1, ex_2, f)$

C and ¬C could then be linearly separable in the m‘-dimensional space

For specific Φ with a kernel function $K(\vec{x}_i, \vec{x}_j) := \Phi(\vec{x}_i)\Phi(\vec{x}_j)$
both training and classification remain efficient,
e.g. for the family of polynoms $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \vec{x}_j + 1)^d$

$\rightarrow$ classification test for new vector $\vec{y}$: $b + \sum_{i=1}^{n} \alpha_i C_i K(\vec{x}_i, \vec{y}) > 0$

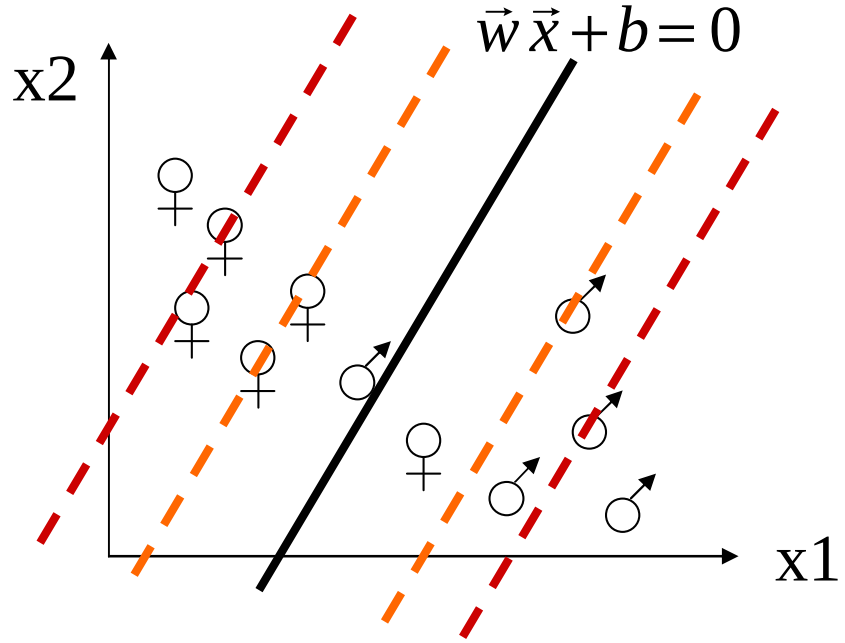
SVM Kernels

Popular and well-understood kernel functions:

- polynomial kernels: $K_{poly}(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \vec{x}_j + 1)^d$
- radial basis function (Gaussian kernel): $K_{RBF}(\vec{x}_i, \vec{x}_j) = \exp((\vec{x}_i - \vec{x}_j)^2 / 2\sigma^2)$
- neural network (sigmoid function): $K_{NN}(\vec{x}_i, \vec{x}_j) = \tanh(\alpha \vec{x}_i \vec{x}_j + \beta)$
- string kernels etc.
  (e.g., for classification of biochemical sequences)

SVMs with "Soft" Separation

If training data are not completely separable
tolerate a few „outliers" on the wrong side of the hyperplane



Find $\vec{w} \in R^m$ and b $\in$ R such that

1. $\|\vec{w}\| + \lambda \sum_{i=1}^{n} \varepsilon_i$ is minimal and
2. $C_i \left(\vec{w}\,\vec{x}_i + b\right) \geq 1 - \varepsilon_i$

for all i=1, ..., n

with control parameter λ for trading off

separation margin $\delta = 1/\|\vec{w}\|$

vs. error sum $\sum_{i=1}^{n} \varepsilon_i$

SVM Engineering

+ Very efficient implementations available
  (e.g., SVM-Light at http://svmlight.joachims.org/):
  with training time empirically found to be
  ≈ quadratic in # training docs (and linear in # features)

+ SVMs can and should usually consider all possible features
(no point for feature selection unless #features intractable)

+ multi-class classification mapped to multiple binary SVMs:
  one-vs.-all or combinatorial design of subset-vs.-complement

– Choice of kernel and soft-margin parameter λ difficult
  and highly dependent on data and application:
  high λ minimizes training error,
  but leads to poor generalization (smaller separation, thus higher risk)

6.6 Hierarchical Classification

given: tree of classes (topic directory) with training data
for each leaf or each node
wanted: assignment of new documents to one or more
leaves or nodes

Top-down approach 1 (for assignment to exactly one leaf):
Determine – from the root to the leaves –
at each tree level the class into which the document suits best.

Top-down approach 2 (for assignment to one or more nodes):
Determine – from the root to the leaves –
at each tree level those classes for which the confidence in
assigning the document to the class lies above some threshold.

Feature Selection for Hierarchical Classification

Features must be good discriminators between classes
with the same parent
→ feature selection must be „context-sensitive“

Examples:

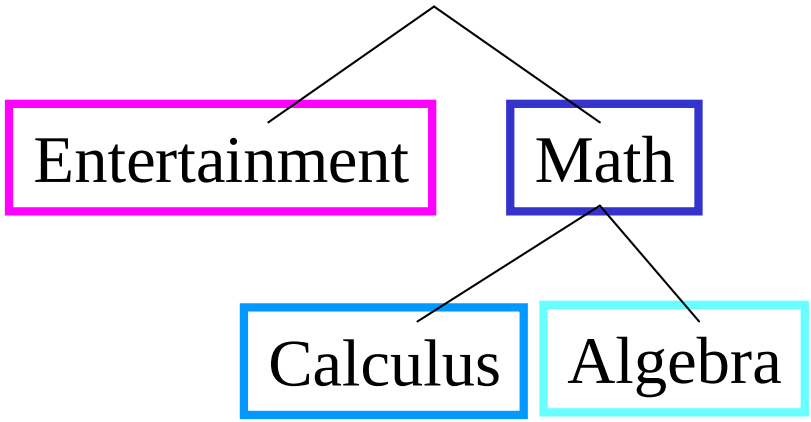
- Terms such as „Definition“, „Theorem“, „Lemma“ are good discriminators
  between Arts, Entertainment, Science, etc.
  or between Biology, Mathematics, Social Sciences, etc.;
  they are poor discriminators
  between subclasses of Mathematics such as Algebra, Stochastics, etc.
- The word „can“ is usually a stopword,
  but it can be an excellent discriminator
  for the topic /Science/Environment/Recycling.

Solution: consider only „competing“ classes with the same parent when using information-theoretic measures for feature selection (see Section 6.2)

Example for Feature Selection

	film	hit	chart	theorem	limit	integral	group	vector
	f1	f2	f3	f4	f5	f6	f7	f8
d1:	1	1	0	0	0	0	0	0
d2:	0	1	1	0	0	0	1	0
d3:	1	0	1	0	0	0	0	0
d4:	0	1	1	0	0	0	0	0
d5:	0	0	0	1	1	1	0	0
d6:	0	0	0	1	0	1	0	0
d7:	0	0	0	0	1	0	0	0
d8:	0	0	0	1	0	1	0	0
d9:	0	0	0	0	0	0	1	1
d10:	0	0	0	1	0	0	1	1
d11:	0	0	0	1	0	1	0	1
d12:	0	0	1	1	1	0	1	0

Class Tree:



training docs:

d1, d2, d3, d4
→ Entertainment

d5, d6, d7, d8
→ Calculus

d9, d10, d11, d12
→ Algebra

Experimental Results on Hierarchical Text Classification (1)

ca. 400 000 documents (from www.looksmart.com)
from ca. 17000 classes in 7 levels:
13 classes at level 1 (Automotive, Business&Finance, Computers&Internet, Entertainment&Media, Health&Fitness, etc.),
150 classes at level 2

ca. 50 000 randomly chosen documents as training data;
for each of the 13+150 classes selection of 1000 terms with the highest mutual information MI(X,C)

Automatic classification of 10 000 documents with SVM (with control parameter λ=0.01):
Top-down assignment of a document to all classes for which the distance to the separating hyperplane was above some threshold δ (with δ experimentally chosen so as to maximize classification quality for training data

$$F1 = \frac{2 * precision * recall}{(precision + recall)}$$)

from: S. Dumais, H. Chen. Hierarchical Classification of Web Content. ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, 2000

Experimental Results on Hierarchical Text Classification (2)

Micro-averaged classification quality (F1 measure)
level 1 (13 classes): F1 ≈ 0.572
level 2 (150 classes): F1 ≈ 0.476

Best and worst classes:

F1 ≈ 0.841	Health & Fitness / Drugs & Medicine
F1 ≈ 0.797	Home & Family / Real Estate
F1 ≈ 0.841	Reference & Education / K-12 Education
F1 ≈ 0.841	Sports & Recreation / Fishing
F1 ≈ 0.034	Society & Politics / World Culture
F1 ≈ 0.088	Home & Family / For Kids
F1 ≈ 0.122	Computers & Internet / News & Magazines
F1 ≈ 0.131	Computers & Internet / Internet & the Web

from: S. Dumais, H. Chen. Hierarchical Classification of Web Content. ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, 2000

Handling Classes with Very Few Training Docs

Problem: classes at or close to leaves may have very few training docs

Idea: exploit feature distributions from ancestor classes

Shrinkage procedure:

- Consider classification test of doc d against class c_n
  with class path c_0 (= root = all docs) c_1 ... c_n
  and assume that classifiers use parameterized probability model
  with (ML) estimators $\theta_{ci,t}$ for class c_i and feature t
- For c_i classifier instead of using $\theta_{ci,t}$
  use „shrunk“ parameters: $\overline{\theta}_{ci,t} = \sum_{i=0}^{n} \lambda_i \theta_{ci,t}$
  where $\sum_{i=0}^{n} \lambda_i = 1$
- Determine λ_i values by iteratively improving accuracy
  on held-out training data

6.7 Classifiers with Semisupervised Learning

Motivation:

- classifier can only be as good as its training data
- and training data is expensive to obtain as it requires intellectual labeling
- and training data is often sparse regarding the feature space

→ use additional unlabeled data to improve the classifier's implicit knowledge of term correlations

Example:

- classifier for topic „cars" has been trained only with documents that contain the term „car" but not the term „automobile"
- in the unlabeled docs of the corpus the terms „car" and „automobile" are highly correlated
- test docs may contain the term „autombobile" but not the term „car"

Simple Iterative Labeling

Let D^K be the set of docs with known labels (training data)
and D^U the set of docs with unknown labels.

Algorithm:
train classifier with $\mathbf{D^K}$ as training data
classify docs in $\mathbf{D^U}$
repeat
    re-train classifier with $\mathbf{D^K}$ and the now labeled docs in D^U
    classify docs in $\mathbf{D^U}$
until labels do not change anymore (or changes are marginal)

Robustness problem:
a few misclassified docs from $\mathbf{D^U}$ could lead
the classifier to drift to a completely wrong labeling

EM Iteration (Expectation-Maximization)

Idea [Nigam et al. 2000]:

E-step: assign docs from $\mathbf{D^U}$ to topics merely with certain probabilities

M-step: use these probabilities to better estimate the model's parameters

Algorithm (for Bayesian classifier):

train classifier with $\mathbf{D^K}$ as training data

E-step: compute probabilities $P[C_k \mid d]$ for all d in $\mathbf{D^U}$

repeat

- ***M-step:*** estimate parameters p_{ik} of the Bayesian model

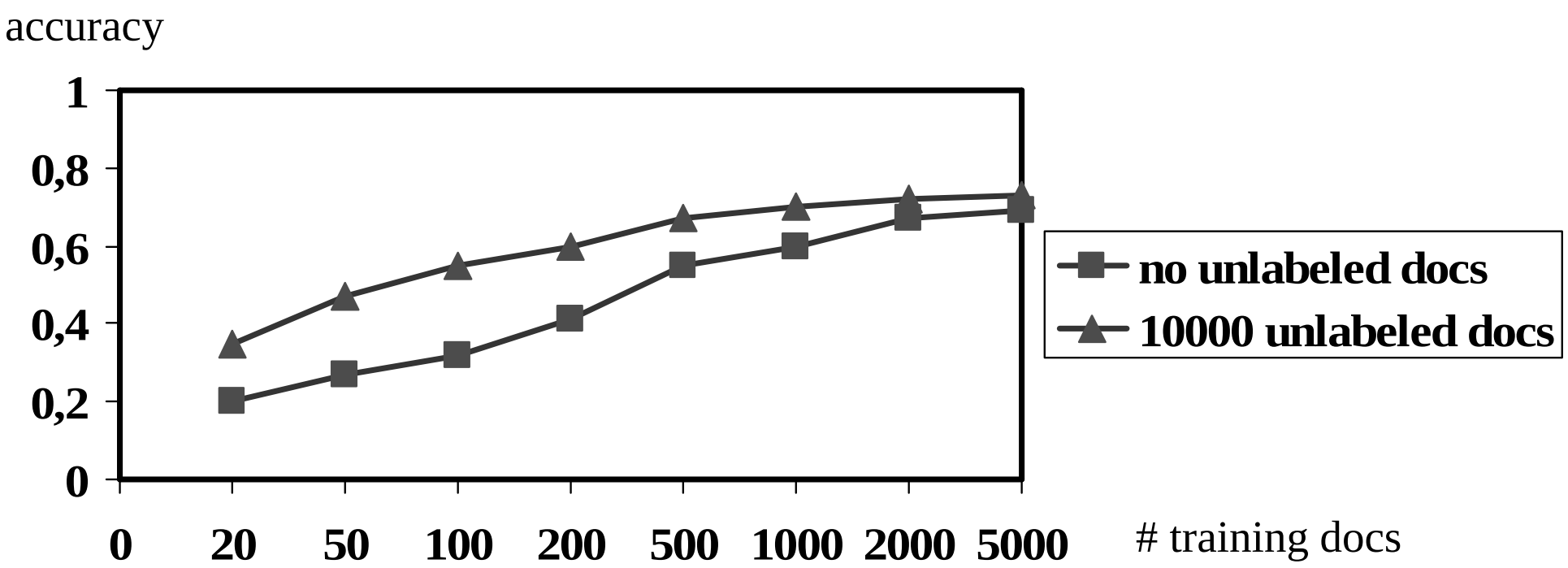
$$p_{ik} = \sum_{d \in C_k} P[C_k \mid d] \cdot tf(t_i, d) \,/\, \sum_{d \in C_k} P[C_k \mid d] \cdot length(d)$$

  (optionally with Laplace smoothing, or using MLE)
- ***E-step:*** recompute probabilities $P[C_k \mid d]$ for all d in $\mathbf{D^U}$

until changes of $\max(P[C_k \mid d] \mid k=1..\#classes)$ become marginal

assign d from $\mathbf{D^U}$ to $\text{argmax}_k(P[C_k \mid d])$

Experimental Results [Nigam et al. 2000]



Co-Training for Orthogonal Feature Spaces

Idea:

- start out with two classifiers A and B for „orthogonal“ feature spaces (whose distributions are conditionally independent given the class labels)
- add best classified doc of A to training set of B, and vice versa (assuming that the same doc would be given the same label by A and B)

Algorithm:

train A and B with orthogonal features of $\mathbf{D^K}$
    (e.g., text terms and anchor terms)
$\mathbf{D^U_A := D^U;\ D^U_B := D^U;\ D^K_A := D^K;\ D^K_B := D^K;}$
repeat
    classify docs in $\mathbf{D^U_A}$ by A and $\mathbf{D^U_B}$ by B
    select the best classified docs from $\mathbf{D^U_A}$ and $\mathbf{D^U_B}$: d_A and d_B
    add d_A to training set $\mathbf{D^K_B}$, add d_B to training set $\mathbf{D^K_A}$
    retrain A using $\mathbf{D^K_A}$, retrain B using $\mathbf{D^K_B}$
until results are sufficiently stable
assign docs from $\mathbf{D^U}$ to classes on which A and B agree

More Meta Strategies

Combine multiple classifiers for more robust results
(usually higher precision and accuracy,
possibly at the expense of reduced recall)

Examples (with m different binary classifiers for class k):

- **unanimous decision:** $C_k(d_j) = 1 \text{ if } \sum_{\nu=1}^{m} C_k^{(\nu)} = m$

- **weighted average:** $C_k(d_j) = 1 \text{ if } \sum_{\nu=1}^{m} \tilde{p}_k^{(\nu)} C_k^{(\nu)} \geq \tau$

  with precision estimator $\tilde{p}_k^{(\nu)}$
  for classifier ν

for further info see machine learning literature
on **ensemble learning (stacking, boosting, etc.)**

6.8 Hypertext Classifiers

Motivation:
the hyperlink neighbors of a test document may exhibit information that helps for the classification of the document

Examples:
- the test document is referenced (only) by a Web page that contains a highly specific keywords that are not present in the document itself (e.g. the word „soccer“ in a page referencing the results of last week‘s Champions League matches)
- the test document is referenced by a Web page that is listed under a specific topic in a manually maintained topic directory (e.g. the topic „sports“ in page referencing the results of ...)
- the test document is referenced by a Web page that also references many training documents for a specific topic

Using Features of Hyperlink Neighbors

Idea: consider terms and possibly class labels of
hyperlink neighbors

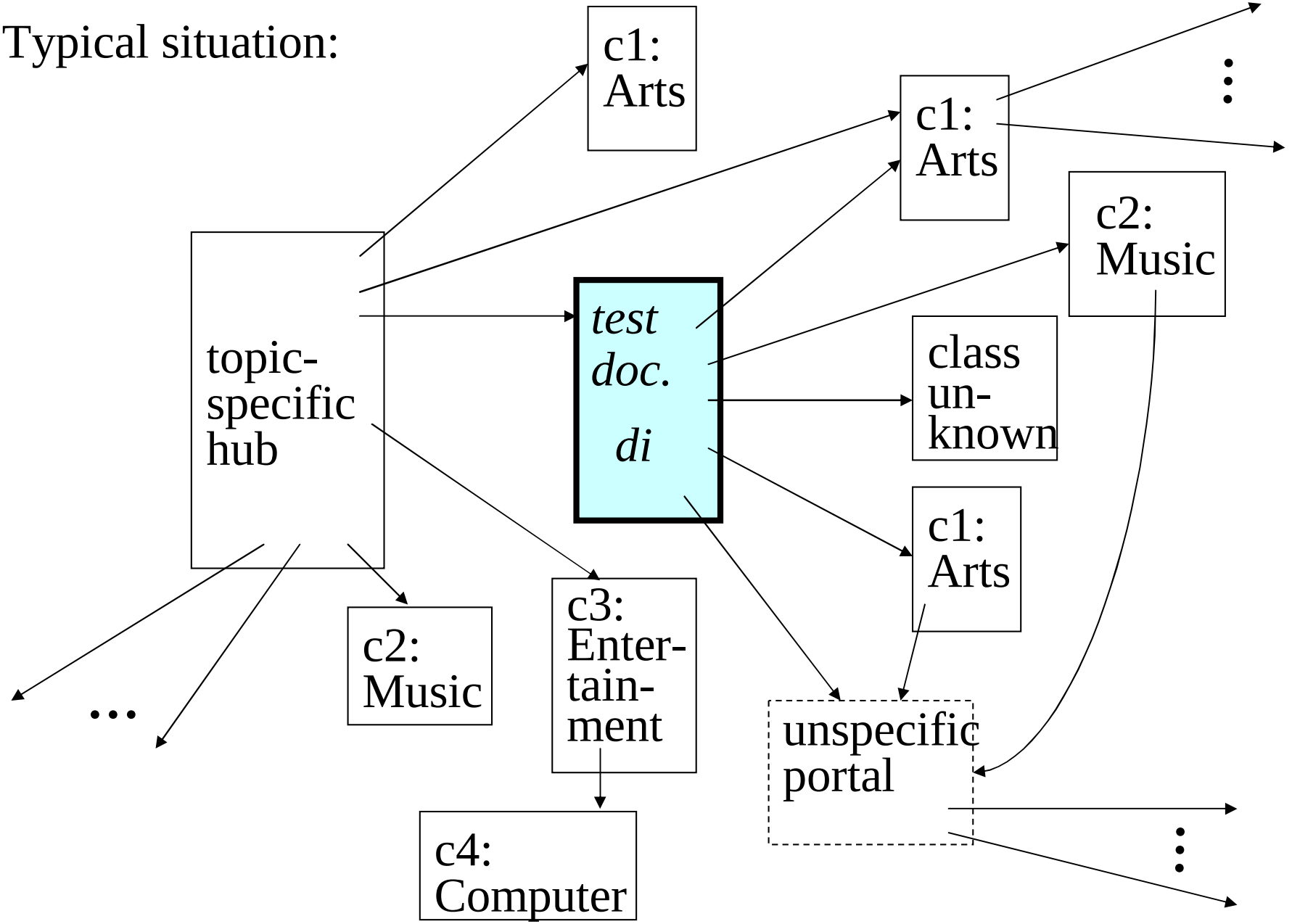
Approach 1:
extend each document by the terms of its neighbors
(within some hyperlink radius ≤ R
and possibly with weights ~ 1/r for hyperlink distance r)

Problem: susceptible to „topic drift“
Example: Link from IRDM course page to www.yahoo.com

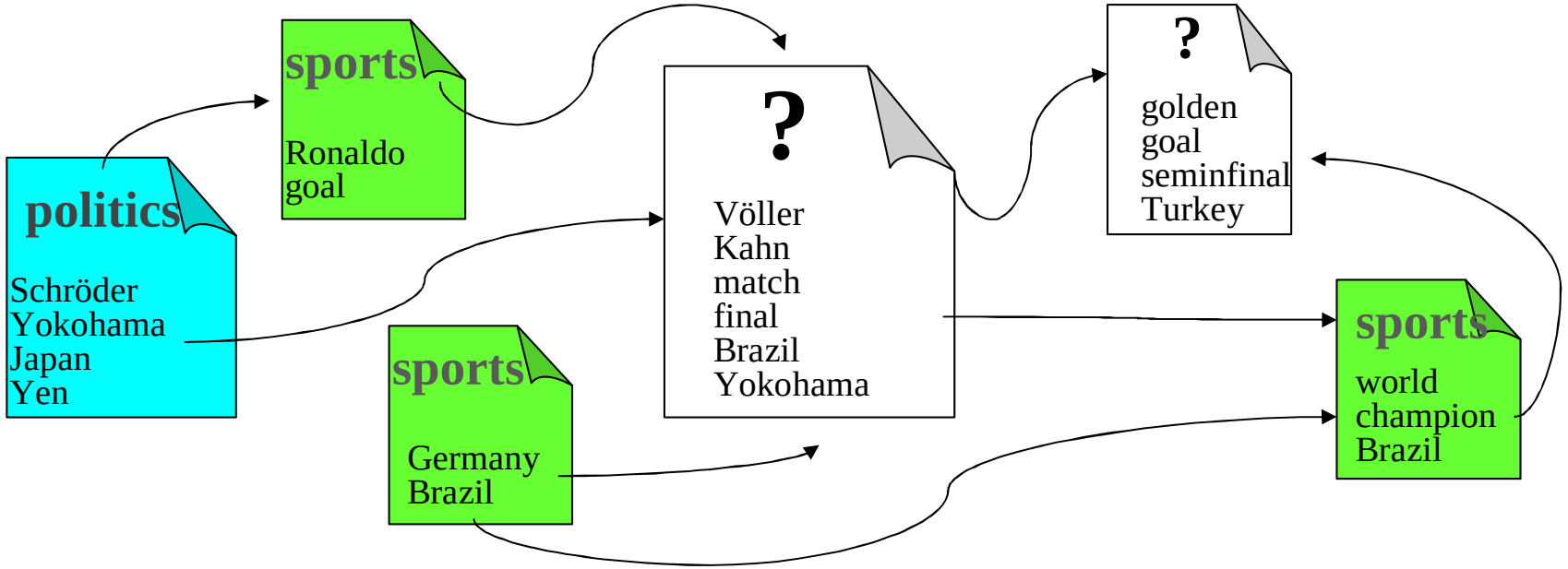
Approach 2:
when classifying a document
consider the class labels of its neighbors

Consideration of Neighbor Class Labels

Typical situation:

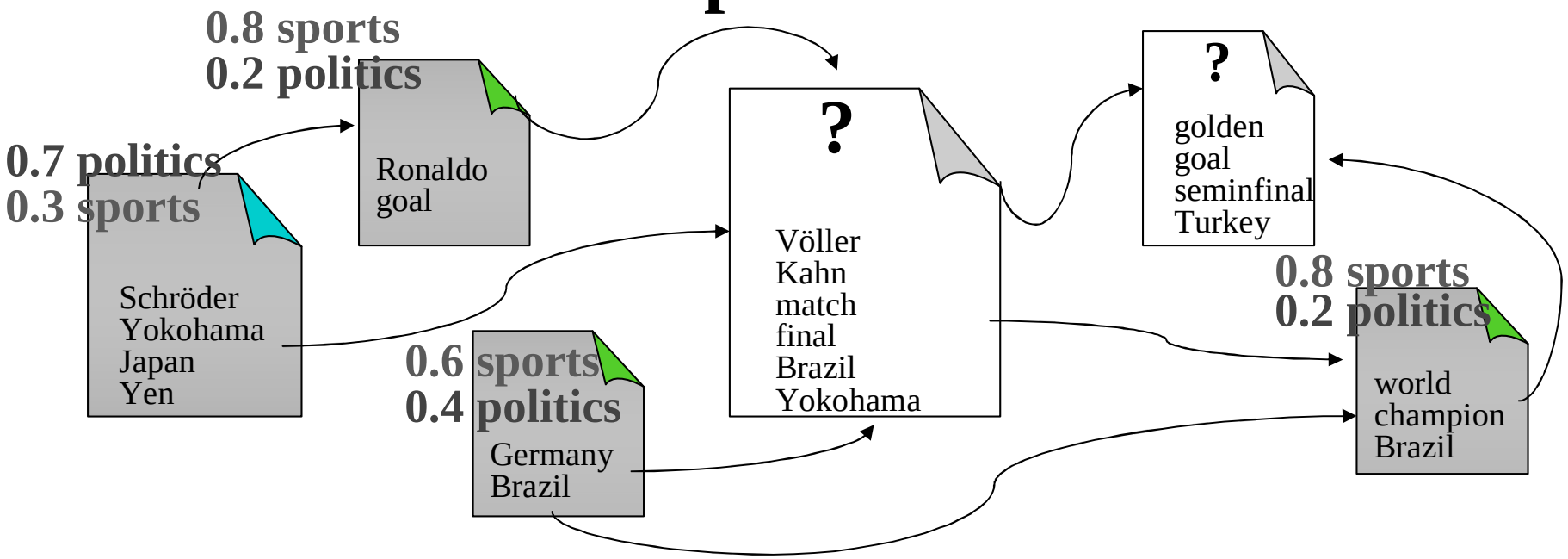


Neighborhood-conscious Feature Space Construction



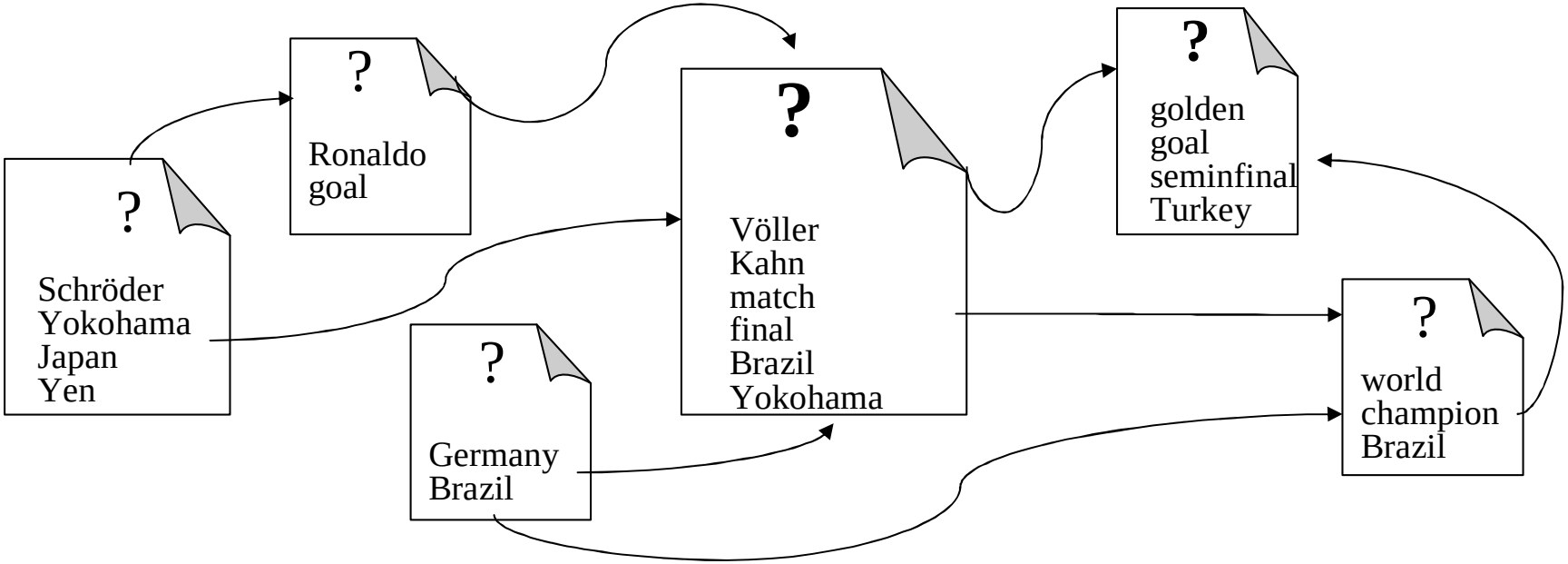
- consider known class labels of neighboring documents

Neighborhood-conscious Feature Space Construction



- consider known class labels of neighboring documents
- consider term-based class probabilities of neighboring documents

Neighborhood-conscious Feature Space Construction

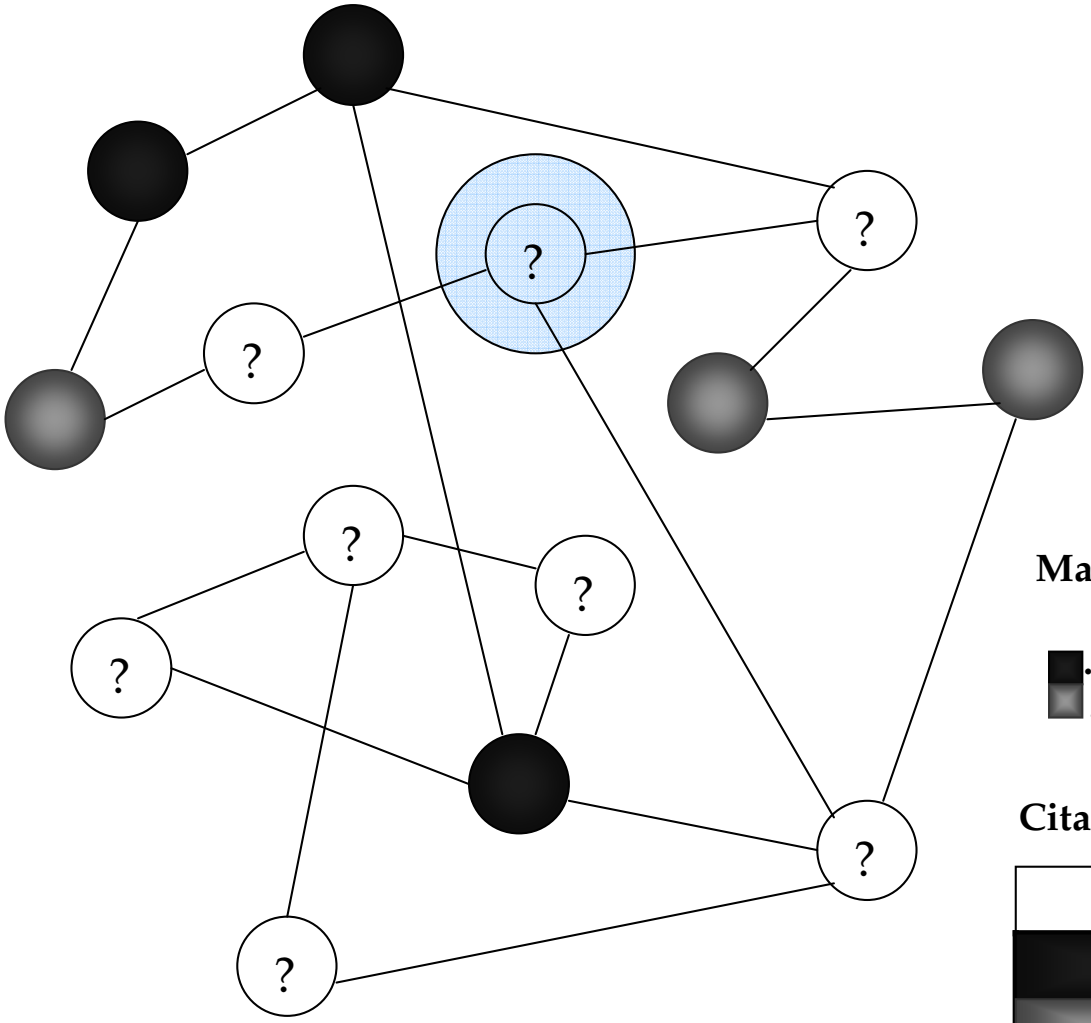


- consider known class labels of neighboring documents
- consider term-based class probabilities of neighboring documents
- evaluate recurrence between class prob. distr. of all documents:

$$P[\, d_i \in C_k \mid N_i^K \,] = \sum_{N_i^U \in \Omega_i} \left(P[\, d_i \in C_k \mid N_i^U, N_i^K \,]\; P[\, N_i^U \mid N_i^K \,] \right)$$

$\rightarrow$ **iterative relaxation labeling for Markov random field**

Relaxation labeling in action



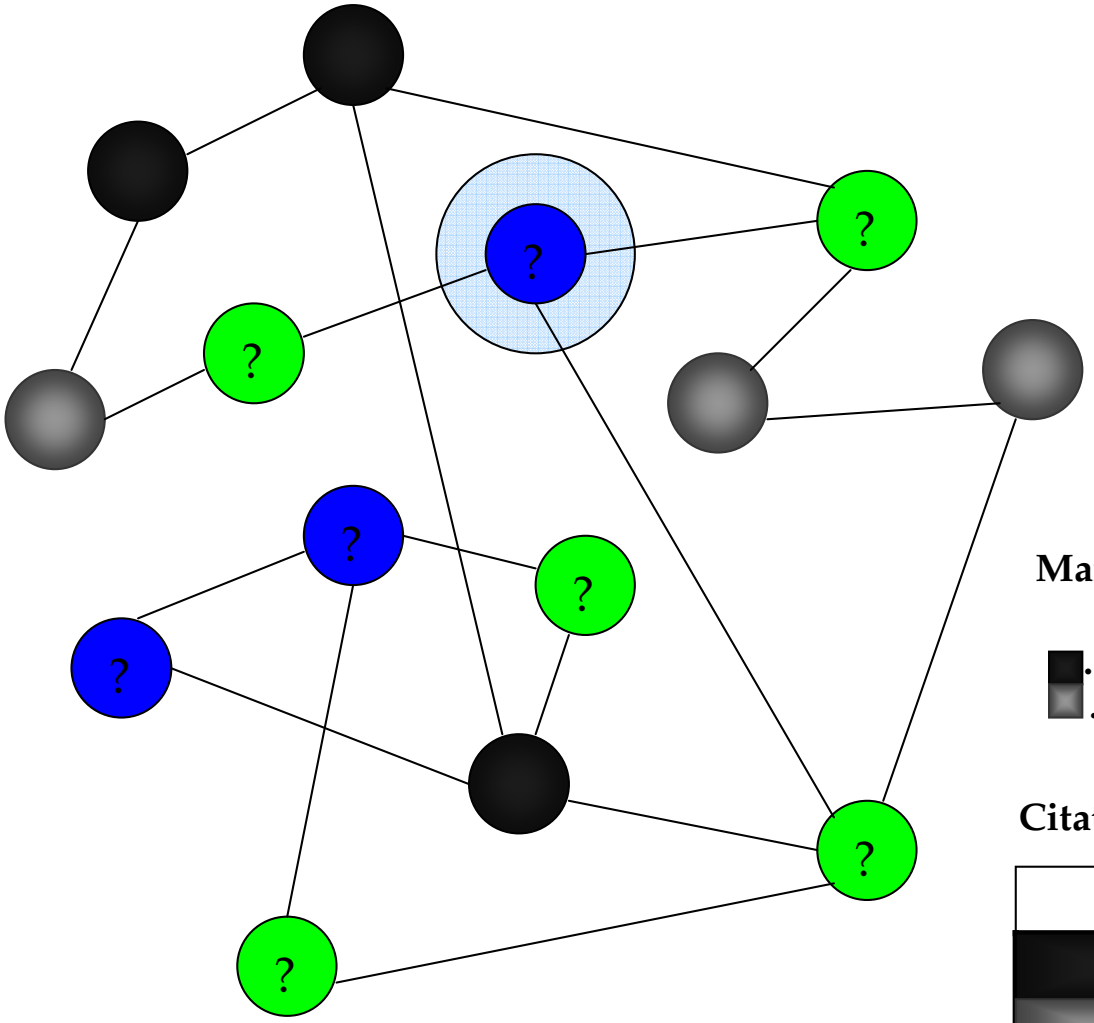
Marginal distributions:

.5 Blue
.5 Green

Citation matrix

	0.75	0.25
	0.38	0.62

Relaxation labeling in action



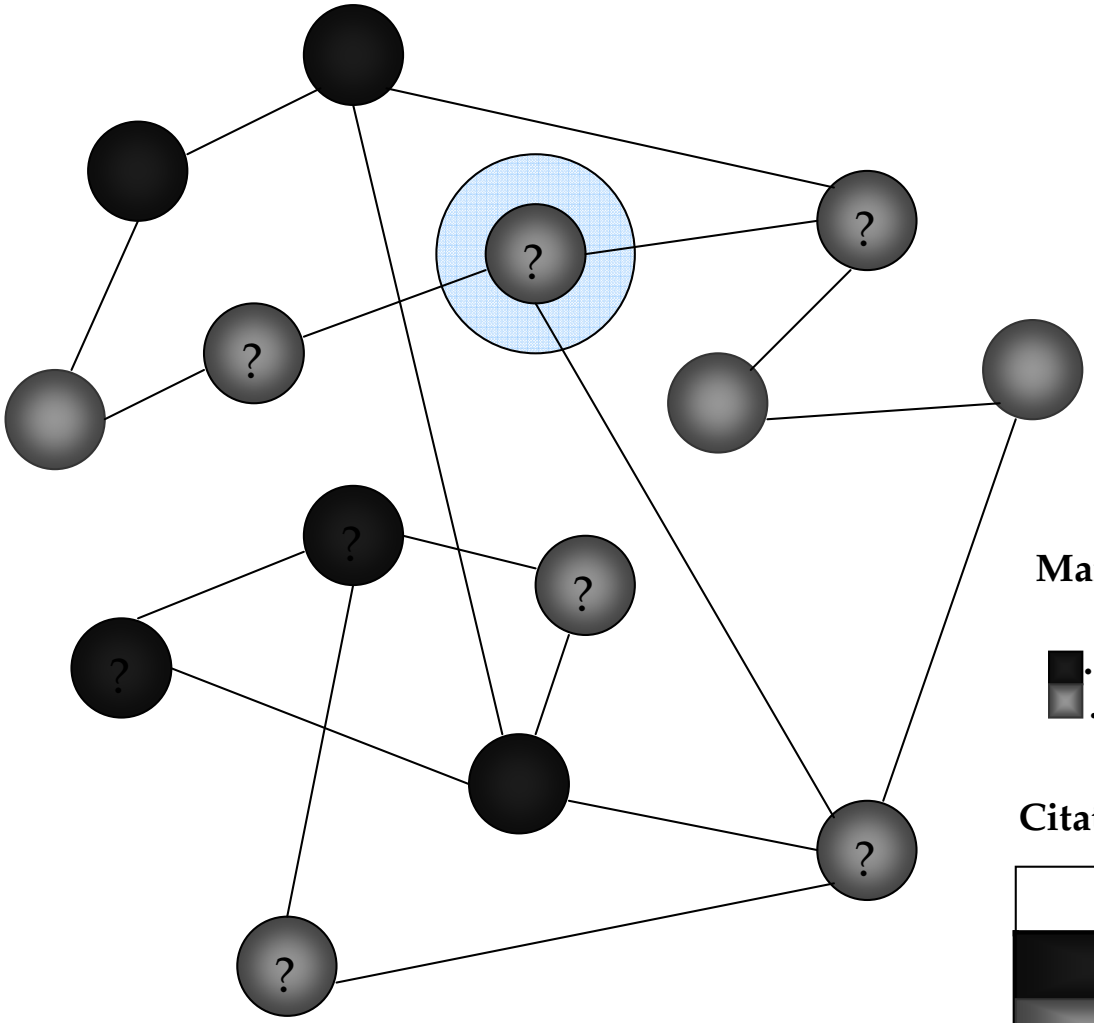
Marginal distributions:

.5 Blue
.5 Green

Citation matrix

	0.75	0.25
	0.38	0.62

Relaxation labeling in action



Marginal distributions:

.5 Blue
.5 Green

Citation matrix

	0.75	0.25
	0.38	0.62

Simple Iterative Labeling with Exploitation of Neighbor Class Labels

for all docs d with unknown class
    classify d using the terms of d
Repeat
- train classifier that exploits class labels of neighbors of all docs d with originally unknown class
- classify d using the extended classifier that exploits neighbor labels

until labels do not change anymore (or changes are marginal)

Naive Bayes Classifier with Consideration of Neighbor Class Labels

analyze $P[\, d_i \in c_k \,|\, d_i \text{ has } \vec{d}_i, \text{ graph } G=(V,E) \text{ on corpus } D\,]$

or $P[\, d_i \in c_k \,|\, d_i \text{ has } \vec{d}_i \text{ and neighbors } N_i \,]$

$$= \frac{P[\vec{d}_i, N_i \,|\, c_k]\, P[c_k]}{P[\vec{d}_i, N_i]} \sim P[\vec{d}_i, N_i \,|\, c_k]\, P[c_k]$$

conditional independence assumptions

$$f(c_k, d_i, N_i) = P[\vec{d}_i, N_i \,|\, c_k] = P[\vec{d}_i \,|\, c_k]\, P[N_i \,|\, c_k]$$

$$= P[\vec{d}_i \,|\, c_k] \prod_{d_j \in pred(d_i)} P[d_j \in c(d_j) \,|\, d_i \in c_k] \prod_{d_j \in succ(d_i)} P[d_j \in c(d_j) \,|\, d_i \in c_k]$$

with pred(di) := {dj | (j, i)∈E} and succ(di) := {dj | (i, j)∈E}

$$= P[\vec{d}_i \,|\, c_k] * \prod_{\nu=1}^{\#classes} P[d_j \in c_\nu \,|\, d_i \in c_k, (j,i) \in E]^{|pred(d_i) \cap c_\nu|}$$

$$* \prod_{\nu=1}^{\#classes} P[d_j \in c_\nu \,|\, d_i \in c_k, (i,j) \in E]^{|succ(d_i) \cap c_\nu|}$$

Digression: Markov Random Fields (MRFs)

Let G = (V, E) be an undirected graph with nodes V and edges E.
The neighborhood N(x) of $x \in V$ is $\{y \in V \mid (x,y) \in E\}$.
All neighborhoods together form a neighborhood system.

Associate with each node $v \in V$ a random variable X_v.
A probability distribution for $(X_{v1}, ..., X_{vn})$ with V={v1, ..., vn}
is called a **Markov random field** w.r.t. the neighborhood system on G
if for all X_{vi} the following holds:

$$P[X_{vi} = x \mid X_{u_1} = x_1 \wedge ... \wedge X_{u_{n-1}} = x_{n-1}\ \}$$
$$= P[X_{vi} = x \mid X_{w_1} = x_1 \wedge ... \wedge X_{w_m} = x_m\ \}$$

$$with\ \{u_1, ..., u_{n-1}\} = V - \{v_i\}\ and\ \{w_1, ..., w_m\} = N(v_i)$$

for MRF theory see, for example, the book:
S.Z. Li, Markov Random Field Modeling in Image Analysis, 2001..

Naive Bayes Classifier with Consideration of Unknown Class Labels of Neighbors (1)

with *known* class labels of neighbors:

assign d_i to class c_k for which $f(c_k, d_i, N_i)$ is maximal

with (partly) *unknown* class labels of neighbors

apply ***Iterative Relaxation Labeling***:

construct neighborhood graph $G_i=(N_i,E_i)$ with radius R around d_i;
classify all docs in N_i based on text features;
repeat
  for all d_j in N_i (incl. d_i) do
    compute the class label of d_j based on
    text features of d_j and the class label distribution of N_j
    using Naive Bayes
  end;
until convergence is satisfactory

Naive Bayes Classifier with Consideration of Unknown Class Labels of Neighbors (2)

Divide neighbor documents of di into

Ni^K – docs with known class labels and

Ni^U – docs with a priori unknown class labels.

Let Δ^K be the union of docs Nj^K with known labels for all considered dj.

Then:

$$P[d_i \in c_k \mid \Delta^K] = \sum_{N_i^U \in \Omega_i} \left(P[di \in c_k \mid N_i^U, \Delta^K] \; P[N_i^U \mid \Delta^K] \right)$$

with the set Ωi of all possible class labelings c of Ni

$$\rightarrow \textit{Iteration } P[d_i \in c_k \mid \Delta^K]^{(p+1)} :=$$

$$\sum_{N_i^U \in \Omega_i} \left(P[di \in c_k \mid N_i^U, N_i^K] * \prod_{d_j \in N_i^U} \left(P[d_j \in c(d_j) \mid \Delta^K]^{(p)} \right) \right)$$

for convergence conditions of Iterative Relaxation Labeling see theory of MRFs

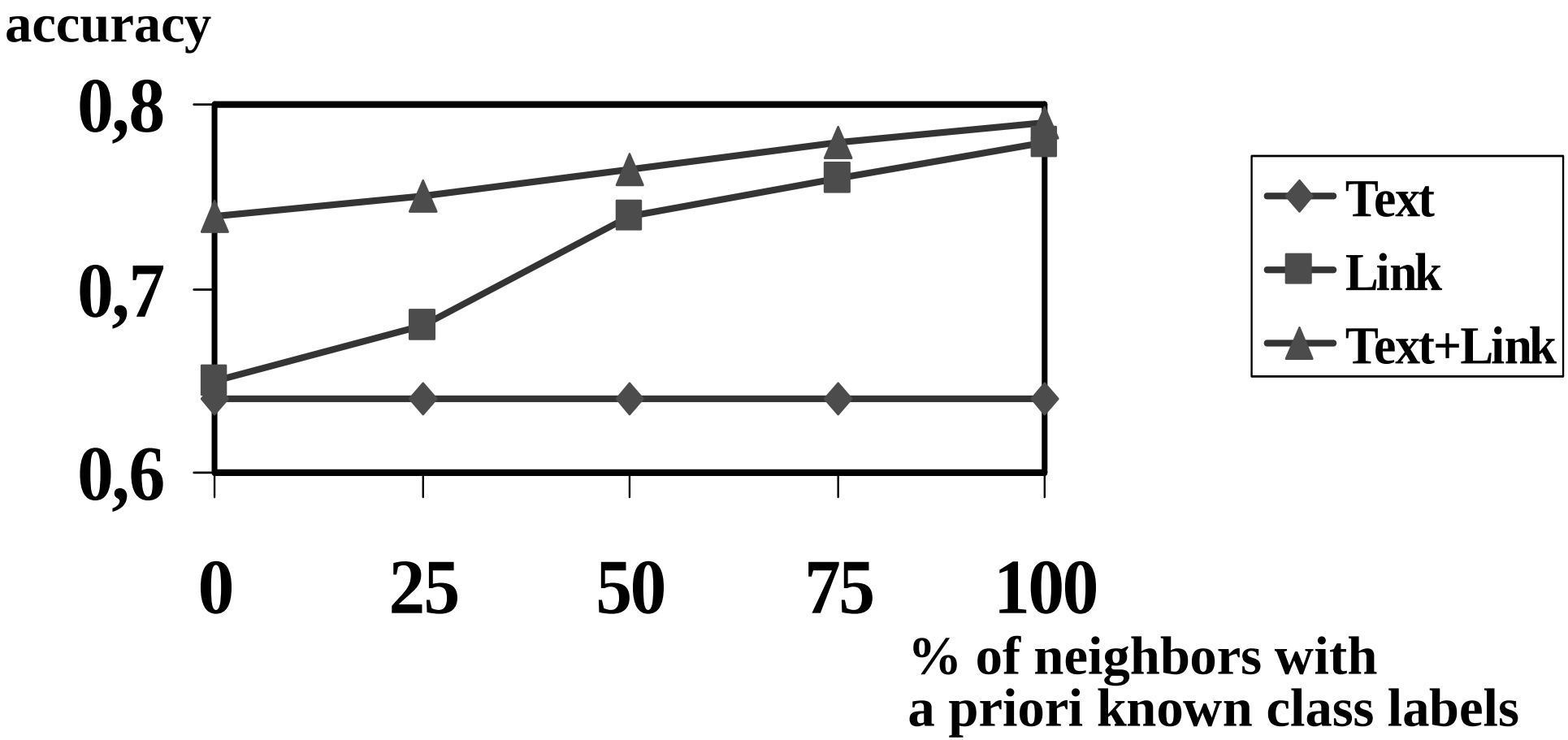
Experimental Results on Hypertext Classification (1)

ca. 1000 patents from 12 classes:
Antenna, Modulator, Demodulator, Telephony, Transmission,
Motive, Regulator, Heating, Oscillator, Amplifier, Resistor, System.
classification accuracy with text features alone: 64%
classification accuracy with hypertext classifier: 79%

ca. 20000 Web documents from 13 Yahoo classes:
Arts, Business&Economy, Computers&Internet, Education,
Entertainment, Government, Health, Recreation, Reference,
Regional, Science, Social Science, Society&Culture.
classification accuracy with text features alone: 32%
classification accuracy with hypertext classifier: 79%

from: S. Chakrabarti, B. Dom, P. Indyk.
Enhanced Hypertext Categorization Using Hyperlinks.
ACM SIGMOD International Conference on Management of Data, Seattle, 1998

Experimental Results on Hypertext Classification (2)



from: S. Chakrabarti, B. Dom, P. Indyk.
Enhanced Hypertext Categorization Using Hyperlinks.
ACM SIGMOD International Conference on Management of Data, Seattle, 1998.

Extended Techniques for Graph-based Classification

- **neighbor pruning:**
  consider only neighbors with content similarity above threshold
  (effectively remove edges)
- **edge weighting:**
  capture confidence in neighbors (e.g. content similarity)
  by edge weights,
  and use weights in class-labeling probabilities
- recompute **neighbor-class-pair probabilities in each RL iteration**
- incorporate **relationship strengths** between different **class labels**

Cost-based Labeling

for simplicity consider only two classes C and $\neg$C

given: ***marginal distribution of classes***: P[d$\in$ X] with X = C or X = $\neg$C
and ***class citation probability distribution:***
P[$d_j \in X$ | d_i references $d_j \wedge d_i \in Y$] with X, Y being C or $\neg$C

find: ***assignment of class labels*** $x_1, ..., x_n$ to documents $d_1, ..., d_n \in D^U$ s.t.
$P[d_1 \in x_1 \wedge ... \wedge d_n \in x_n \mid D^K]$ is maximized

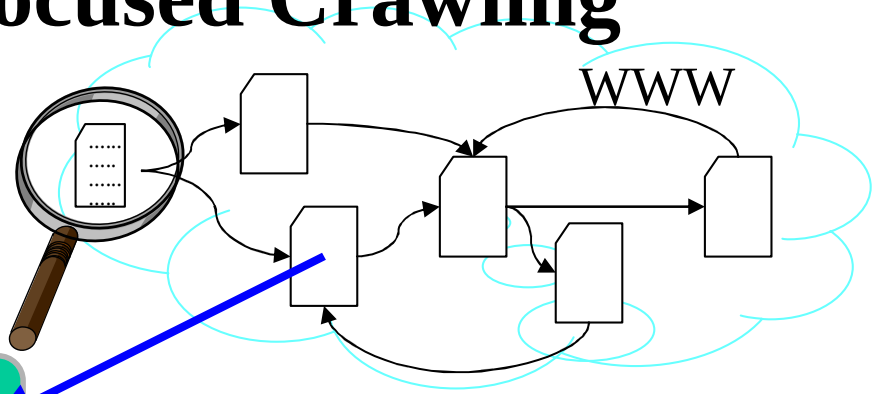
approach: minimize $-\log P[d_1 \in x_1 \wedge ... \wedge d_n \in x_n \mid D^K]$
$= -\Sigma_i \log P[d_i \in x_i \mid D^K]$ assuming independence

$\rightarrow$ NP-complete problem, but with good approximation algorithms.

see: J. Kleinberg, E. Tardos, Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields, Journal of the ACM Vol.49 No.5, 2002.

11.9 Application: Focused Crawling

automatially build
personal topic directory



Bookmarks

Semistructured Data

Web Retrieval

Soumen Chakrabarti
Susan Dumais Homepage
SIGIR 2000 TECHNICAL PROGRAM SCHEDULE
Byron Dom's home page
http://www.almaden.ibm.com/almaden/feat/www8/
Weiyi Meng's Home page
Towards a Highly-Scalable and Effective Metasearch Engine
http://www.henzinger.com/monika/mpublications.html
The Anatomy of a Search Engine

Data Mining

Johannes Gehrke's Publications
Data Mining and Knowledge Discovery Table of Contents
Knowledge Discovery in Databases and Data Mining
. . . . . .

seeds

training

Crawler

Classifier

Link Analysis

Root

DB Core Technology

Semistrutured Data

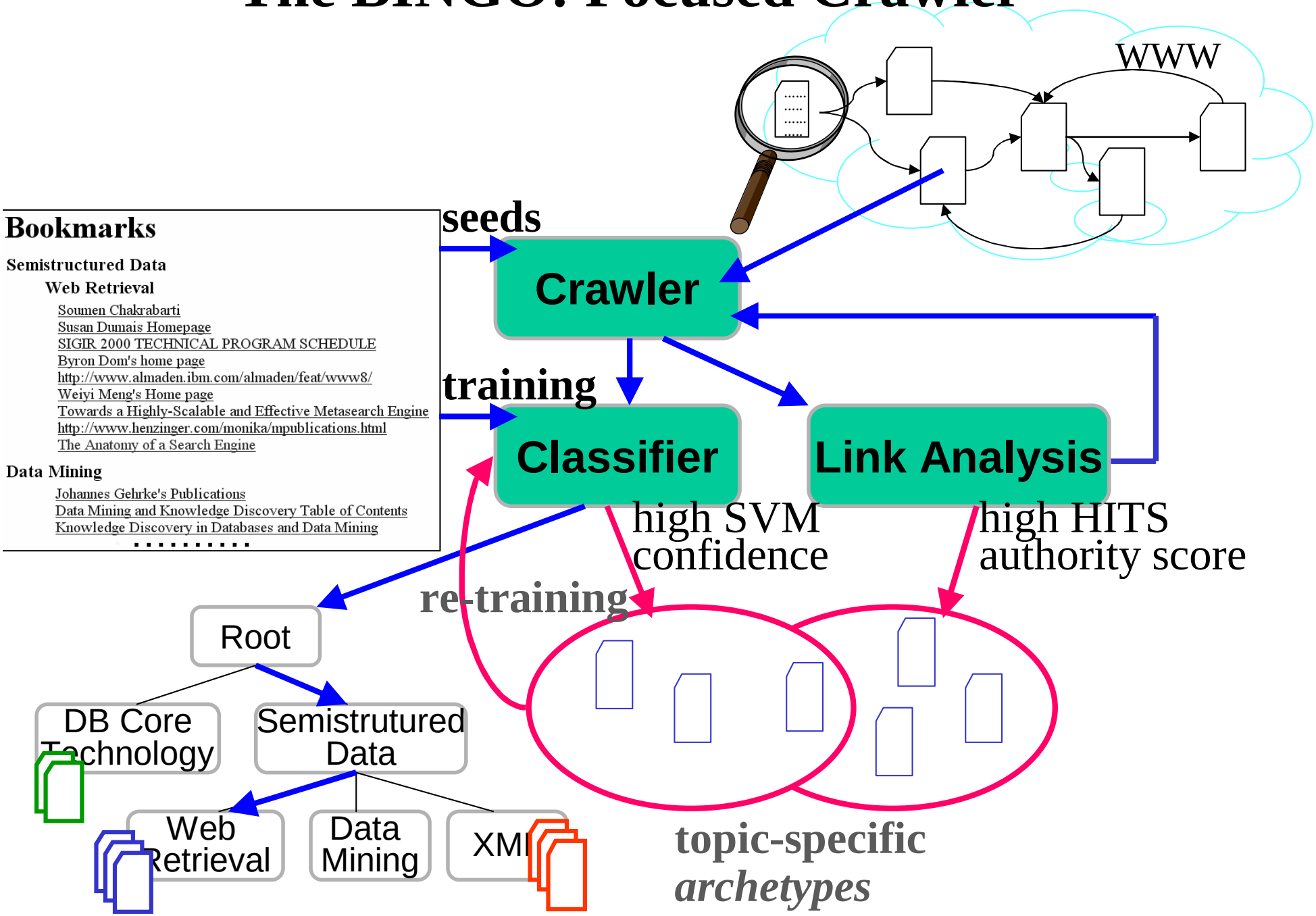
Web etrieval

Data Mining

XML

critical issues:
- **classifier accuracy**
- **feature selection**
- **quality of training data**

The BINGO! Focused Crawler



BINGO! Adaptive Re-training and Focus Control

for each topic V do {

***archetypes(V)* := top docs of *SVM confidence* ranking**

$\cup$ top *authorities* of V ;

remove from archetypes(V) all docs d that do not satisfy

***confidence(d)* $\geq$ *mean confidence(V)* ;**

recompute feature selection based on *archetypes(V)* ;

recompute SVM model for V with *archetypes(V)* as training data }

combine re-training with two-phase crawling strategy:

- ***learning phase:***

  aims to find archetypes (*high precision*)

  $\rightarrow$ hard focus for crawling vicinity of training docs

- ***harvesting phase:***

  aims to find results (*high recall*)

  $\rightarrow$ soft focus for long-range exploration with tunnelling

Summary of Chapter 6

+ Automatic classification has numerous applications
+ Naive Bayes, decision trees, SVMs are mature methods
- Danger of overfitting: aim for balance between training error and generalization
  → may require feature selection
  or tuning of regularization parameters
- Semisupervised classifiers aim to address training bottleneck
- Model selection (parameters, feature engineering) is crucial
- Graph-awareness is promising form of richer features

Additional Literature for Chapter 6

Classification and Feature-Selection Models and Algorithms:

- S. Chakrabarti, Chapter 5: Supervised Learning
- C.D. Manning / H. Schütze, Chapter 16: Text Categorization, Section 7.2: Supervised Disambiguation
- J. Han, M. Kamber, Chapter 7: Classification and Prediction
- T. Mitchell: Machine Learning, McGraw-Hill, 1997, Chapter 3: Decision Tree Learning, Chapter 6: Bayesian Learning, Chapter 8: Instance-Based Learning
- D. Hand, H. Mannila, P. Smyth: Principles of Data Mining, MIT Press, 2001, Chapter 10: Predictive Modeling for Classification
- M.H. Dunham, Data Mining, Prentice Hall, 2003, Chapter 4: Classification
- M. Ester, J. Sander, Knowledge Discovery in Databases, Springer, 2000, Kapitel 4: Klassifikation
- Y. Yang, J. Pedersen: A Comparative Study on Feature Selection in Text Categorization, Int. Conf. on Machine Learning, 1997
- C.J.C. Burges: A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery 2(2), 1998
- S.T. Dumais, J. Platt, D. Heckerman, M. Sahami: Inductive Learning Algorithms and Representations for Text Categorization, CIKM Conf. 1998

Additional Literature for Chapter 6

Advanced Topics (Semisupervised C., Graph-aware C., Focused Crawling, MDL, etc.):

- S. Chakrabarti, Chapter 6: Semisupervised Learning
- K. Nigam, A. McCallum, S. Thrun, T. Mitchell: Text Classification from Labeled and Unlabeled Data Using EM, Machine Learning 39, 2000
- S. Chakrabarti, B. Dom, P. Indyk: Enhanced Hypertext Categorization Using Hyperlinks, ACM SIGMOD Conference 1998
- S. Chakrabarti, M. van den Berg, B. Dom: Focused crawling: a new approach to topic-specific Web resource discovery, WWW Conf., 1999
- S. Sizov et al.: The BINGO! System for Information Portal Generation and Expert Web Search, CIDR Conference, 2003
  S. Siersdorfer, G. Weikum: Using Restrictive Classification and Meta Classification for Junk Elimination, ECIR 2005
- M. H. Hansen, B. Yu: Model Selection and the Principle of Minimum Description Length, Journal of the American Statistical Association 96, 2001
- P.Grünwald, A Tutorial introduction to the minimum description length principle Advances in Minimum Description Length, MIT Press, 2005