An Overview of WHIRL, or
Matching Almost Anything Quickly: How and Why

William Cohen
Carnegie Mellon University

My background

- Rutgers, 1986-1990: Explanation based learning (learning from examples and prior knowledge)
- AT&T Bell Labs/AT&T Research, 1990-2000:
  - Learning logic programs/description logics
  - What representations work well for learners?
  - Scalable rule learning (RIPPER system)
  - Text categorization/information extraction
  - WHIRL (this talk)

My background

- WhizBang Labs, April 2000-May 2002
  - More text categorization, IE, matching
  - "Wrapper induction": learning to extract data from a single web site (Cohen *et al*, WWW-2002)
  - Improving text classifiers by recognizing structure of "index pages" (Cohen, NIPS-2002)
- Carnegie Mellon's CALD center: last year
  - Information extraction from on-line biomedical publications: subcellular location information from text and images
  - Evaluating aspect retrieval systems
  - Privacy issues related to data integration

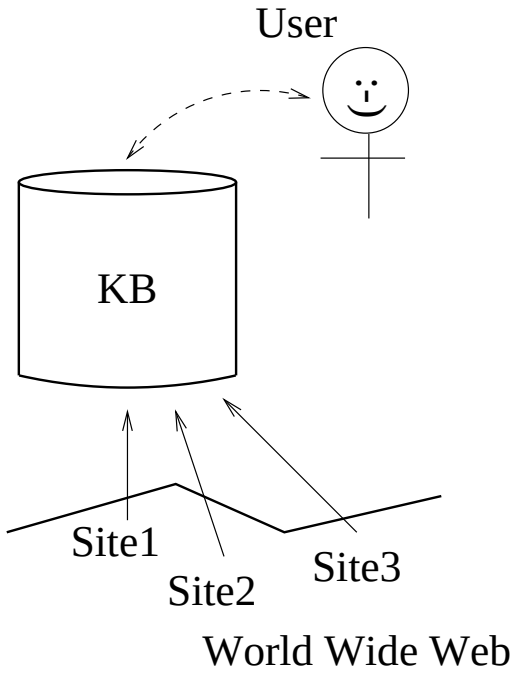
Grand visions for information access

- The semantic web: a world-wide database as widely distributed, fluid, and easy to extend as the web.
- Peer-to-peer databases: exchange and query structured information across thousands of client/server machines.
- Large-scale information extraction: extract a database of structured information from thousands or millions of documents.
- Large-scale information integration, e.g. across deep-web sources: make thousands of databases look like one.
- The "world wide knowledge base": make the existing web look like a single huge knowledge base.

A common thread: merging structured data

Notice: planning people see a planning problem, learning people see a learning problem, programming language people see a language problem, ...

The real problem is representation.



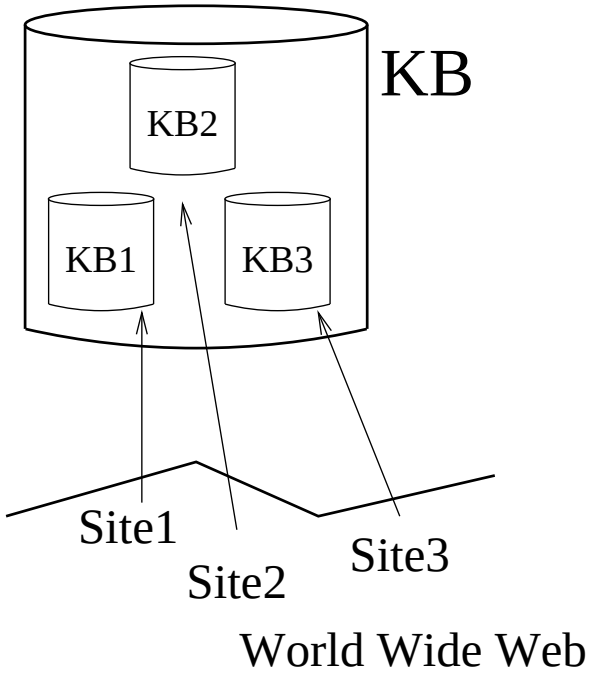
What's the research problem?

Clarification: There are two kinds of information systems:

1. Search engines, clipping services, hypertext, ... store and deliver potentially relevant documents to a user.
   - Easy to handle information from diverse sources.
2. Databases, KR systems, ... store facts and perform deduction on behalf of a user.
   - Very hard to handle information from diverse sources.

What's the research problem?

We don't know how to reason with information that comes from many different, autonomous sources.



all mallards are waterfowl	+	`duck.jpg` is a picture of a mallard	=	`duck.jpg` is a picture of a waterfowl

Taxonomy

Order	Species
waterfowl	mallard
waterfowl	bufflehead
raptor	osprey
raptor	bald eagle
...	...

+

Images

Species	File
robin	robin.jpg
mallard	duck.jpg
osprey	hawk.jpg
penguin	tweety.jpg
...	...

=

Order	Species	File
waterfowl	mallard	duck.jpg
raptor	osprey	hawk.jpg
...	...	...

mallards are found in New Jersey	+	`duck.jpg` is a picture of a mallard	=	`duck.jpg` is a picture of something found in New Jersey

NJ Birds

Species
robin
mallard
osprey
...

+

Images

Species	File
robin	robin.jpg
mallard	duck.jpg
penguin	tweety.jpg
...	...

=

Species	File
robin	robin.jpg
mallard	duck.jpg
...	...

Deduction enables modularity.

Why deduction requires co-operation

```
-? nj_bird(X),image(X,File).
nj_bird(mallard). nj_bird(robin). ...
image(mallard,'duck.jpg'). image(american_robin,'robin.jpg'). ...
```

The providers of the nj_bird and image facts have to agree on:

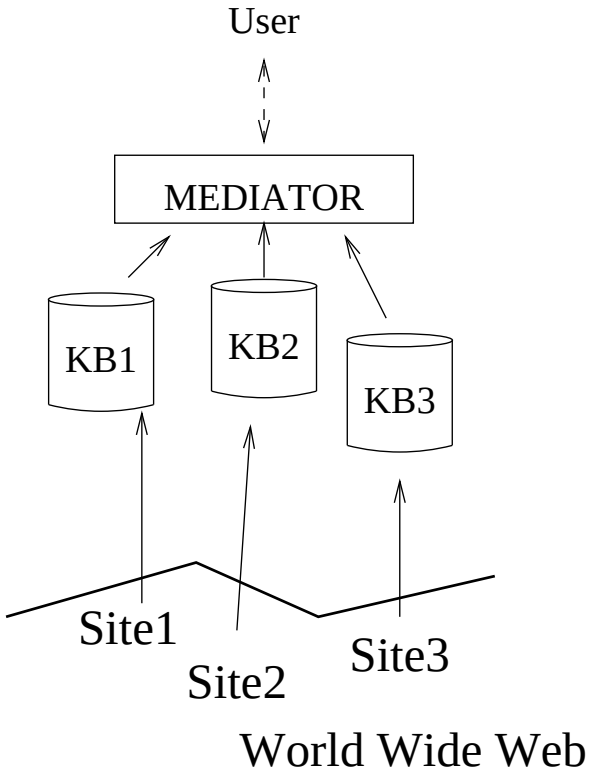
- predicate names and argument positions (schema);
- taxonomic information;
- formal names (OIDs) for every entity they describe;
- ...

Deduction without co-operation

If information providers don't co-operate, then a "mediator" program must translate:

'robin' → 'american_robin'

How hard is it to determine if two names refer to the same thing?



Humongous	Humongous Entertainment	Microsoft	Microsoft Kids Microsoft/Scholastic
Headbone	Headbone Interactive		
			American Kestrel
The Lion King: Storybook	Lion King Animated StoryBook	Kestrel	Eurasian Kestrel
		Canada Goose	Goose, Aleutian Canada
Disney's Activity Center, The Lion King	The Lion King Activity Center	Mallard	Mallard, Mariana

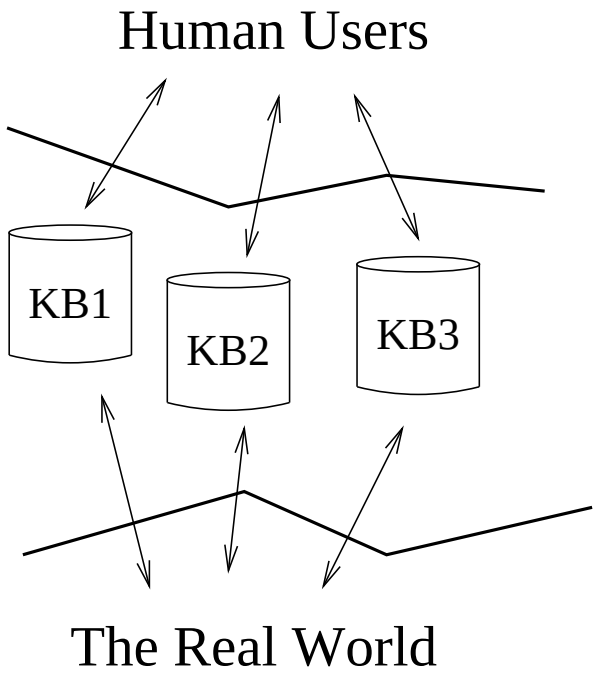
Bell Labs	AT&T Bell Labs
AT&T Research	AT&T Labs
Bell Telephone Labs	AT&T Labs—Research
AT&T Labs–Research, Shannon Laboratory	Lucent Innovations Bell Labs Technology

Conclusion: name-coreference is an AI-complete problem.

What's the research problem?

We need a general means for integrating formally unconnected knowledge bases.

We must exploit these facts: the individual KB's model the same real world, and communicate with the same users.



The WHIRL approach

Key points:

- Use informal names and descriptions as object identifiers.
- Use techniques from information retrieval (IR) to guess if two descriptions refer to the same object.
- Use soft ($\approx$ probabilistic) reasoning for deduction.

Formal reasoning methods over informally identified objects.

Overview of WHIRL

- WHIRL (Word-based Heterogeneous Information Representation Language) is somewhere between IR systems (document delivery) and KR systems (deduction).
- Outline:
  - Data model: how information is stored.
  - WHIRL query language
  - Accuracy results
  - Key ideas for implementation
  - Efficiency results
  - More results and conclusions

Background: Information retrieval

Ranked retrieval: (e.g., Altavista, Infoseek, ...) given a query Q, find the documents $d_1, \ldots, d_r$ that are **most similar** to Q.

Similarity of d_i and d_j is measured using set of terms T_{ij} common to d_i and d_j:

$$SIM(d_i, d_j) = \sum_{t \in T_{ij}} weight(t, d_i) \cdot weight(t, d_j)$$

- A **term** is a single word (modulo stemming, ...)
- Heuristic: make $weight(t, d)$ large if t is frequent in d, or if t is rare in the corpus of which d is an element.

Background: Information retrieval

Similarity of d_i and d_j is measured using set of terms T_{ij} common to d_i and d_j:

$$SIM(d_i, d_j) = \sum_{t \in T_{ij}} weight(t, d_i) \cdot weight(t, d_j)$$

- Heuristic: make $weight(t, d)$ large if t is frequent in d (TF), or if t is rare in the corpus of which d is an element (IDF).
- Example: if the corpus is a list of company names:
  - Low weight: "Inc", "Corp", ...
  - High weight: "Microsoft", "Lucent", ...
  - Medium weight: "Acme", "American", ...

Background: Information retrieval

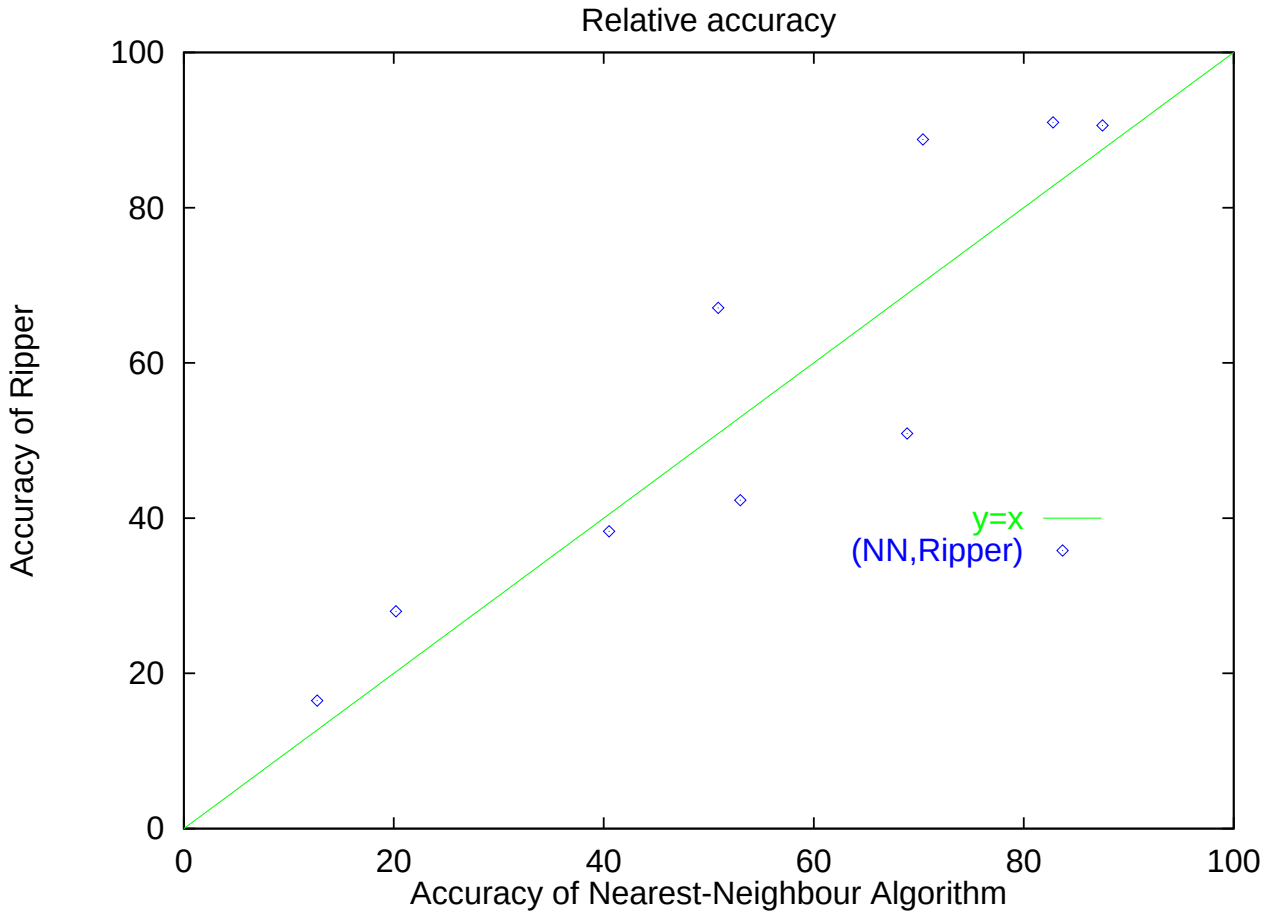
It's notationally convenient to think of a document d_i as a long, sparse vector, v_i.

If $\vec{v}_i = \langle v_{i,1}, \ldots, v_{i,|T|} \rangle$, $v_{i,t} = weight(t, d_i)$, and $||v_i|| = 1$:

$$\begin{aligned} SIM(d_i, d_j) &= \sum_{t \in T} weight(t, d_i) \cdot weight(t, d_j) \\ &= \vec{v}_i \cdot \vec{v}_j \end{aligned}$$

Also, $0 \leq SIM(d_i, d_j) \leq 1$.

Effectiveness of the TF-IDF "vector space" representation



Cinema	Movie	Show Times
Roberts Theaters Chatham	Brassed Off	7:15 - 9:10
Berkeley Cinema	Hercules	4:15 - 7:30
Sony Mountainside Theater	Men In Black	7:40 - 8:40 - 9:30 - 10:10

listing($\vec{v}_{RTC}, \vec{v}_{BO}, \vec{v}_{T79}$), 1.
listing($\vec{v}_{BC}, \vec{v}_{H}, \vec{v}_{T47}$), 1.
listing($\vec{v}_{SMT}, \vec{v}_{MIB}, \vec{v}_{T789}$), 1.

review($\vec{w}_{MIB97}, \vec{w}_{R1}$), 1.
review($\vec{w}_{FO}, \vec{w}_{R2}$), 1.
review($\vec{w}_{SB}, \vec{w}_{R3}$), 1.

Each $\vec{v}_i, \vec{w}_i$ is a document vector.
Each fact has a score $s \in [0, 1]$.

Movie	Review
Men in Black, 1997	$(***)$ One of the biggest hits of ...
Face/Off, 1997	$(** \frac{1}{2})$ After a slow start, ...
Space Balls, 1987	$(*\frac{1}{2})$ Not one of Mel Brooks' best efforts, this spoof ...

$\vec{v}_{MIB} = \langle \ldots, v_{black}, \ldots, v_{in}, \ldots, v_{men}, \ldots \rangle$
$\vec{w}_{MIB97} = \langle \ldots, w_{black}, \ldots, w_{in}, \ldots, w_{men}, \ldots, w_{1997}, \ldots \rangle$
$w_{1997} \approx 0 \implies sim(\vec{v}_{MIB}, \vec{w}_{MIB97}) \approx 1$

Queries in WHIRL

- Syntax: WHIRL = (similarity)
  Prolog − function symbols − recursion − negation + X∼Y
- Semantics (details in Cohen, SIGMOD98):
  - A ground formula gets a score $s \in [0, 1]$
  - $\text{Score}(p(a_1, \ldots, a_k)) = s$ for DB literals.
  - $\text{Score}(a \sim b) = SIM(a, b)$ for similarity literals.
  - $\text{Score}(\phi \wedge \psi) = \text{Score}(\phi) \cdot \text{Score}(\psi)$.
  - $\text{Score}(\phi \vee \psi) = 1 - (1 - \text{Score}(\phi))(1 - \text{Score}(\psi))$
  - Answer to a query Q is an ordered list of the r substitutions $\theta_1, \ldots, \theta_r$ that give $Q\theta_i$ the highest scores.
    (User provides r).

Queries in WHIRL

- Syntax: WHIRL =
  unions of conjunctive SQL queries + X∼Y
- Semantics (details in Cohen, SIGMOD98):

  SELECT $\quad \mathrm{r}_{i_1}.\mathrm{f}_{i_1}, \mathrm{r}_{i_2}.\mathrm{f}_{i_2}, \ldots$

  FROM $\quad \mathrm{R}_1$ as $\mathrm{r}_1, \mathrm{R}_2$ as $\mathrm{r}_2, \ldots, \mathrm{R}_k$ as r_K

  WHERE $\quad \phi(\mathrm{R}_1, \ldots, \mathrm{R}_K)$

  – Answer is an ordered list of tuples.

  – A tuple is defined by binding each r_i to a tuple $t_j = \langle a_{j,1}, \ldots, a_{j,\ell} \rangle \in \mathrm{R}_i$, and then SELECT-ing the appropriate fields.

  – Answer: the n tuples with max score for ϕ (and t_j's).

- Score$(a \sim b) = SIM(a, b)$ for similarity literals.
- Score$(\phi \wedge \psi) =$ Score$(\phi) \cdot$ Score(ψ).
- Score$(\phi \wedge \psi) =$ Score$(\phi) \cdot$ Score(ψ).
- Score$(\phi \vee \psi) = 1 -$ (1 $-$ Score$(\phi))(1 -$ Score$(\psi))$
- Score for $r_i \rightarrow t_j$ is taken from DB score for t_j.
- Final score: Score$(\phi) \cdot \Pi_i Score(r_i \rightarrow t_j)$

Sample WHIRL queries

Standard ranked retrieval:

"find reviews of sci-fi comedies".

?- review(Title,Text) $\wedge$ Text$\sim$"sci-fi comedy"

FROM review as r SELECT * WHERE r.text$\sim$"sci-fi comedy"

(score 0.22): $\theta_1 = \{\text{Title}/\vec{w}_{MIB97}, \text{Text}/\vec{w}_{R1}\}$

(score 0.19): $\theta_2 = \{\text{Title}/\vec{w}_{SB}, \text{Text}/\vec{w}_{R4}\}$

(score 0.13): $\theta_2 = \ldots$

Sample WHIRL queries

Standard DB queries: *"find reviews for movies playing in Mountainside"* (assume single-term "movie IDs" in DB)

?- review(Id1,T1,Text) $\wedge$ listing(C,Id2,T2,Time)
$\wedge$ Id1$\sim$Id2 $\wedge$ C$\sim$"Sony Mountainside Theater"
FROM review as r,listing as l SELECT *
WHERE r.id=l.id AND l.cinema$\sim$"Sony Mountainside Theater""

(score 1.00): $\theta_1 = \{\text{Id1}/\vec{v}_{93}, \text{Id2}/\vec{w}_{93}, \text{Text}/\vec{w}_{R1}, \ldots\}$
(score 1.00): $\theta_2 = \ldots$

Cinema	Id	Movie	Time
...	21	Brassed Off	...
Sony ...	93	Men In Black	...

Id	Movie	Review
93	Men in Black, 1997	...
44	Face/Off, 1997	...

Sample WHIRL queries

Mixed queries: *"where is [Men in Black] playing?"*

?- review(Id1,T1,Text) $\wedge$ listing(C,Id2,T2,Time)
$\wedge$ Id1$\sim$Id2 $\wedge$ Text$\sim$"sci-fi comedy with Will Smith"
FROM review as r,listing as l SELECT *
WHERE r.id=l.id AND r.text$\sim$"sci-fi comedy with Will Smith"

(score 0.22): $\theta_1 = \{\text{Id1}/\vec{v}_{93}, \text{Id2}/\vec{w}_{93}, \text{Text}/\vec{w}_{R1}, \ldots\}$
(score 0.13): $\theta_2 = \ldots$

Cinema	Id	Movie	Time
...	21	Brassed Off	...
Sony ...	93	Men In Black	...

Id	Movie	Review
93	Men in Black, 1997	...
44	Face/Off, 1997	...

A realistic situation

Cinema	Movie	Show Times
Roberts Theaters Chatham	Brassed Off	7:15 - 9:10
Berkeley Cinema	Hercules	4:15 - 7:30
Sony Mountainside Theater	Men In Black	7:40 - 8:40 - 9:30 - 10:10

With real Web data, there will be no common ID fields, only informal names.

Movie	Review
Men in Black, 1997	$(***)$ One of the biggest hits of ...
Face/Off, 1997	$(** \frac{1}{2})$ After a slow start, ...
Space Balls, 1987	$(*\frac{1}{2})$ Not one of Mel Brooks' best efforts, this spoof ...

Sample WHIRL queries

"Similarity" joins: *"find reviews of movies currently playing"*

?- review(Title1,Text) $\wedge$ listing(_,Title2,Time) $\wedge$ Title1$\sim$Title2

FROM review as r,listing as l SELECT *

WHERE r.title$\sim$l.title

(score 0.97): $\theta_1 = \{$ Title1/$\vec{v}_{MIB}$, Title2/$\vec{w}_{MIB97}, \ldots\}$

(Men in Black) (Men in Black, 1997)

...

(score 0.41): $\theta_2 = \{$ Title1/$\vec{v}_{BO}$, Title2/$\vec{w}_{FO}, \ldots\}$

(Brassed Off) (Face/Off)

...

How well do similarity joins work?

?- top500(X), hiTech(Y), X∼Y

FROM top500,hiTech SELECT * WHERE top500.name∼hiTech.name

top500:	*hiTech:*
Abbott Laboratories	ACC CORP
Able Telcom Holding Corp.	ADC TELECOMMUNICATION INC
Access Health, Inc.	ADELPHIA COMMUNICATIONS CORP
Acclaim Entertainment, Inc.	ADT LTD
Ace Hardware Corporation	ADTRAN INC
ACS Communications, Inc.	AIRTOUCH COMMUNICATIONS
ACT Manufacturing, Inc.	AMATI COMMUNICATIONS CORP
Active Voice Corporation	AMERITECH CORP
Adams Media Corporation	APERTUS TECHNOLOGIES INC
Adolph Coors Company	APPLIED DIGITAL ACCESS INC
...	APPLIED INNOVATION INC
	...

Sample company-name pairings

WHIRL output on business.html

Evaluating similarity joins

- **Input:** query
- **Output:** ordered list of documents

1	√	a_1	b_1
2	√	a_2	b_2
3	×	a_3	b_3
4	√	a_4	b_4
5	√	a_5	b_5
6	√	a_6	b_6
7	×	a_7	b_7
8	√	a_8	b_8
9	√	a_9	b_9
10	×	a_{10}	b_{10}
11	×	a_{11}	b_{11}
12	√	a_{12}	b_{12}

Precision at K: G_K/K

Recall at K: G_K/G

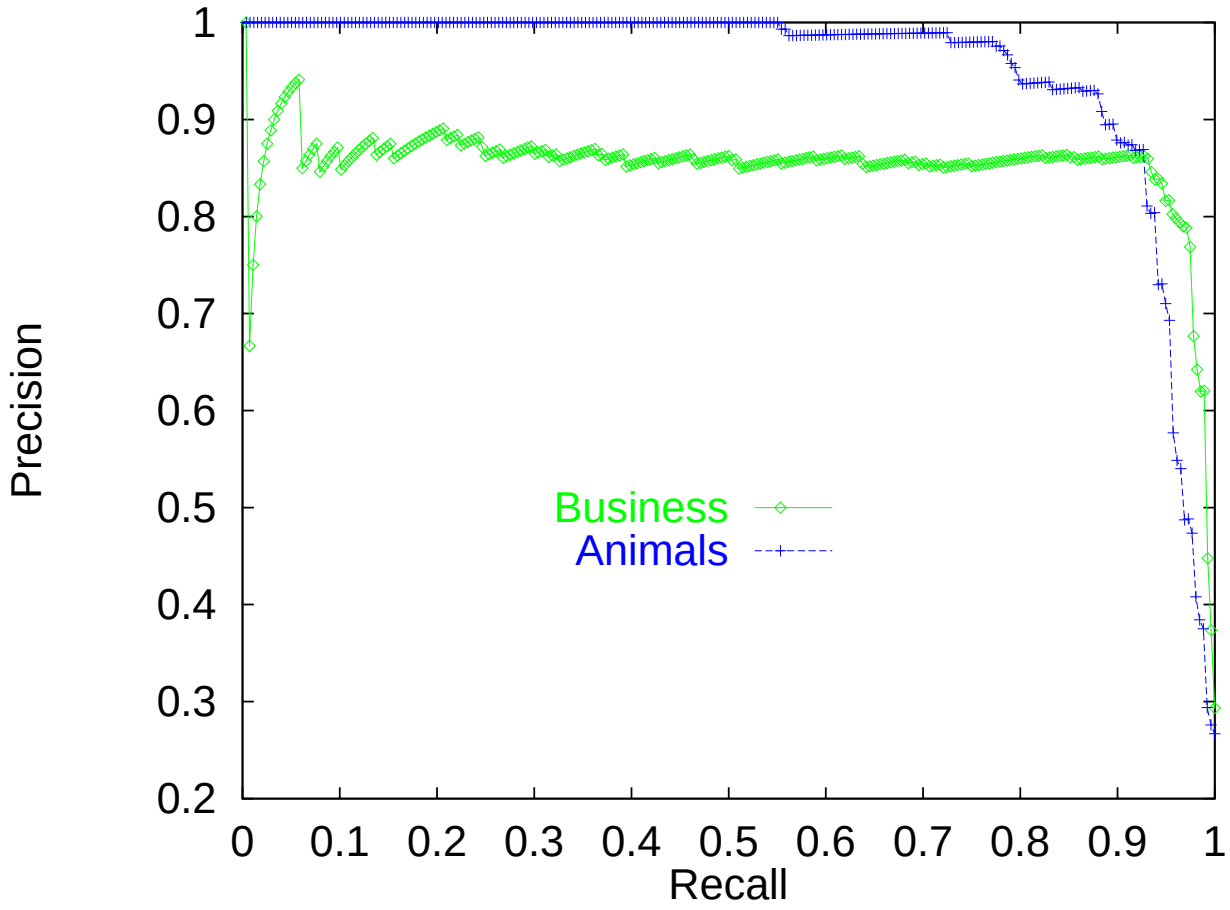
G: # good pairings

G_K: # good pairings in first K

Evaluating similarity joins

- Pick relations p, q with > 2 plausible keys
- Perform "similarity join" using first key field
- Mark a pairing correct ("relevant") if secondary key matches
- Compute precision and recall over first 1000 rankings
- Examples
  - Business: company name, web site
  - Animals: common name, scientific name
  - etc

Evaluating similarity joins



Evaluating WHIRL queries

Additional experiments:

- Repeat with more datasets from more domains.
  - Average precision ($\approx$ area under precision-recall curve) ranges from 85% to 100% over 13 joins in 6 domains.
- Repeat for more complex join queries.
  - Average precision drops from 94% for 2-way joins to 90% for 5-way joins (averaged over many queries in one domain).
- Evaluate other things to do with WHIRL.
- How can you implement WHIRL efficiently?

An efficient implementation

Key ideas for current implementation:

- Central problem: given Q, find best substitution.
  - Currently, using $A*$ search.
- Search space: partial substitutions.

  e.g., for "?- r(X),s(Y),X∼Y", possible state is $\{X = \vec{x}\}$.
- Key operator: when Q contains "$\vec{x}$∼Y", find good candidate bindings for Y quickly.
  - Use inverted indices.

An efficient implementation

- Key step: state is a substitution θ, $Q\theta$ contains "s(Y),$\vec{x}\sim$Y". Need to find good candidate bindings for Y quickly.
  1. Pick some term t with large weight in $\vec{x}$.
  2. Use inverted index to get

$$I_{t,s,1} = \{\vec{y} : s(\vec{y}) \in \text{DB and } y_t > 0\}$$

- To compute heuristic value of state, use fact that

$$score(\vec{x} \sim Y) \;\leq\; \max_{\vec{z} \in I_{t,s,1}} (\sum_t x_t \cdot z_t) \;\leq\; \sum_t x_t \cdot (\max_{\vec{z} \in I_{t,s,1}} z_t)$$

- Indexing and bounds well-known in IR
  (Buckley-Lewitt, Turtle-Flood's *maxscore* alg)

An efficient implementation

- Controlled experiments: for 2-relation soft joins WHIRL is:
  - about 20x faster than naive use of inverted indices
  - from 4-10x faster than Turtle-Flood's *maxscore*
- In practice, for typical queries to two real web-based integration systems:
  - Game domain: 15 sites, 23k+ tuples, avg 0.3sec/query
  - Birding domain: 35 sites, 140k+ tuples, avg 0.2sec/query

The extraction problem

Sometimes it's difficult to extract even an informal name from its context:

- Fox Interactive has a fully working demo version of the Simpsons Cartoon Studio. (Win and Mac)
- Vividus Software has a free 30 day demo of Web Workshop (web authoring package for kids!) Win 95 and Mac
- Scarlet Tanager (58kB) *Piranga olivacea*. New Paltz, June 1997. "...Robin-like but hoarse (suggesting a Robin with a sore throat)." (Peterson) "..a double-tone which can only be imitated by strongly humming and whistling at the same time." (Mathews)

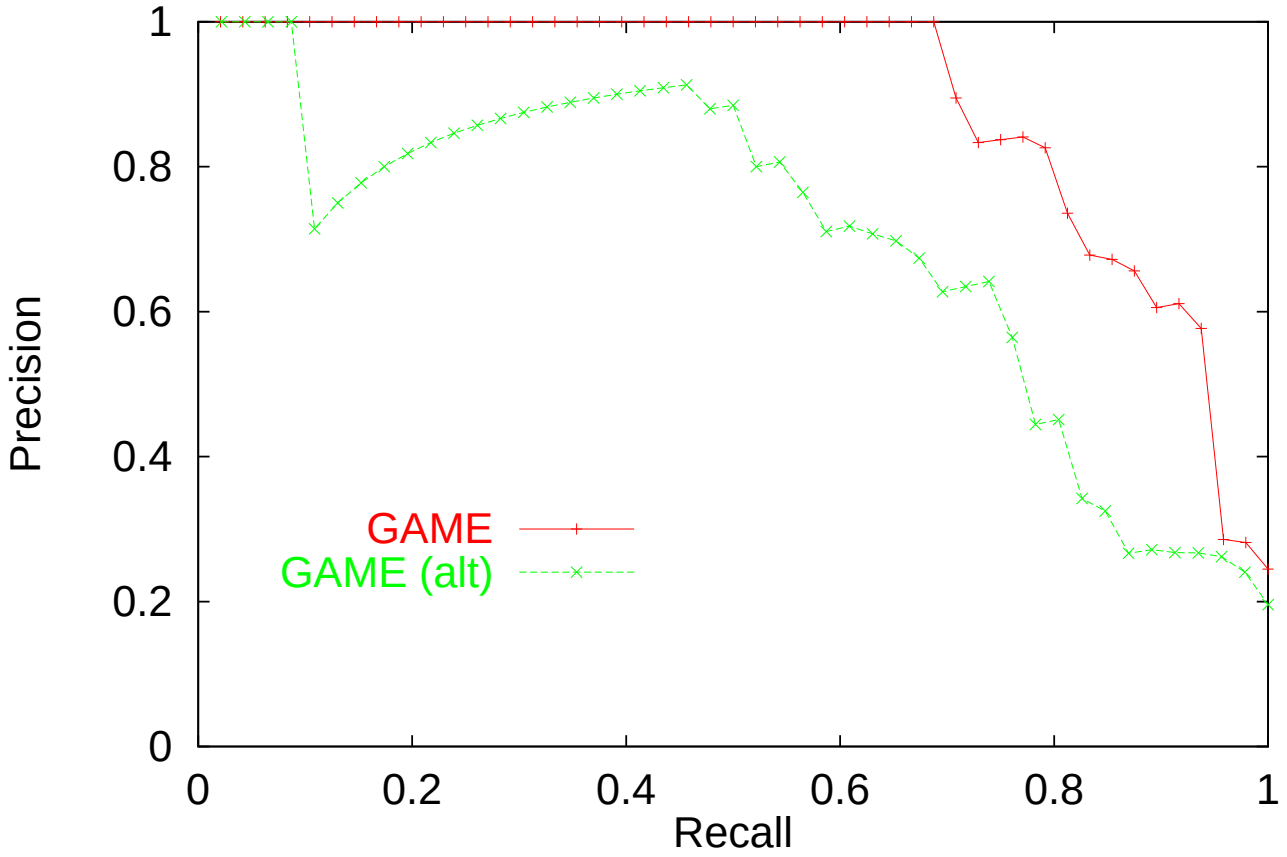
The extraction problem

Idea: use text without trying to extract names.

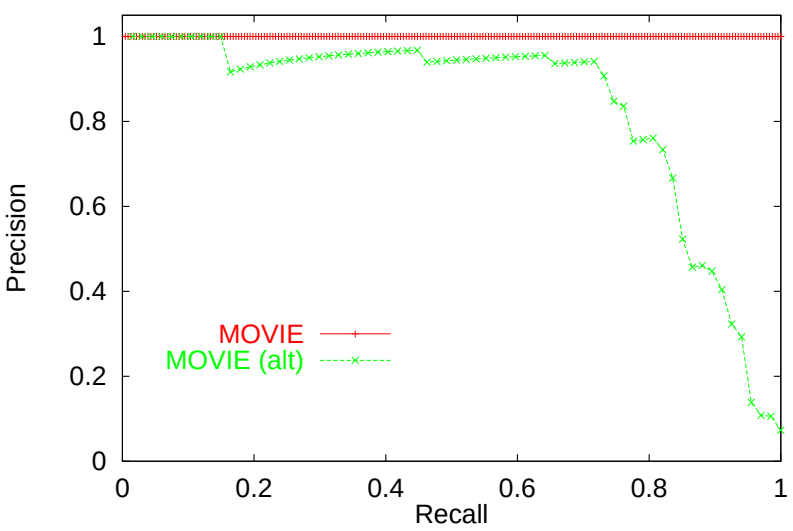
?- paragraph(X),name(Y),X∼Y

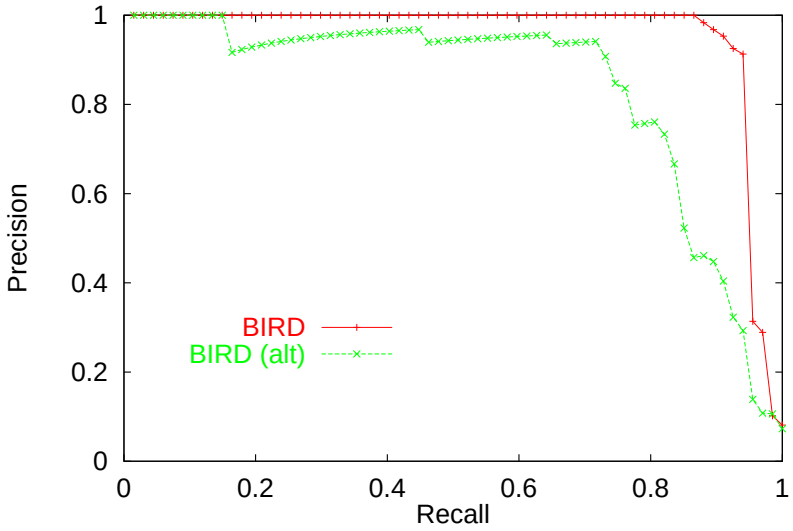
80.26	Ubi Software has a demo of Amazing Learning Games with Rayman.	Amazing Learning Games with Rayman	✓
78.25	Interplay has a demo of Mario Teaches Typing. (PC)	Mario Teaches Typing	✓
75.91	Warner Active has a small interactive demo for Where's Waldo at the Circus and Where's Waldo? Exploring Geography (Mac and Win)	Where's Waldo? Exploring Geography	✓
74.94	MacPlay has demos of Marios Game Gallery and Mario Teaches Typing. (Mac)	Mario Teaches Typing	✓
71.56	Interplay has a demo of Mario Teaches Typing. (PC)	Mario Teaches Typing 2	×

Deduction without extraction



Deduction without extraction





Movie 1: full review (no extraction).

Movie 2: movie name, cinema name & address, showtimes.

More uses of WHIRL: Classification?

review("Putt-Putt Travels Through Time", url1).
category("Putt-Putt's Fun Pack", "adventure").
category("Time Traveler CD", "history").
...
"find me reviews of adventure games"
v(Url) ←
review(Game1,Url) ∧ category(Game2,Cat)
∧ Game1∼Game2 ∧ Cat∼"adventure"

To answer this query, WHIRL guesses the class "adventure" based on similarities between names.

More uses of WHIRL: Classification

$$\text{category(Cat)} \leftarrow \text{test(X)} \wedge \text{train(Y,Cat)} \wedge \text{X} \sim \text{Y}$$

- Here train contains a single unclassified example, and test contains a set of training examples with known categories. (from Cohen&Hirsh, KDD-98)
- WHIRL here performs a sort of K-NN classification.
  1. Find r best bindings for X,Y,Cat
  2. Combine evidence using noisy-or:
     $\text{Score}(\phi \wedge \psi) = \text{Score}(\phi) \cdot \text{Score}(\psi)$

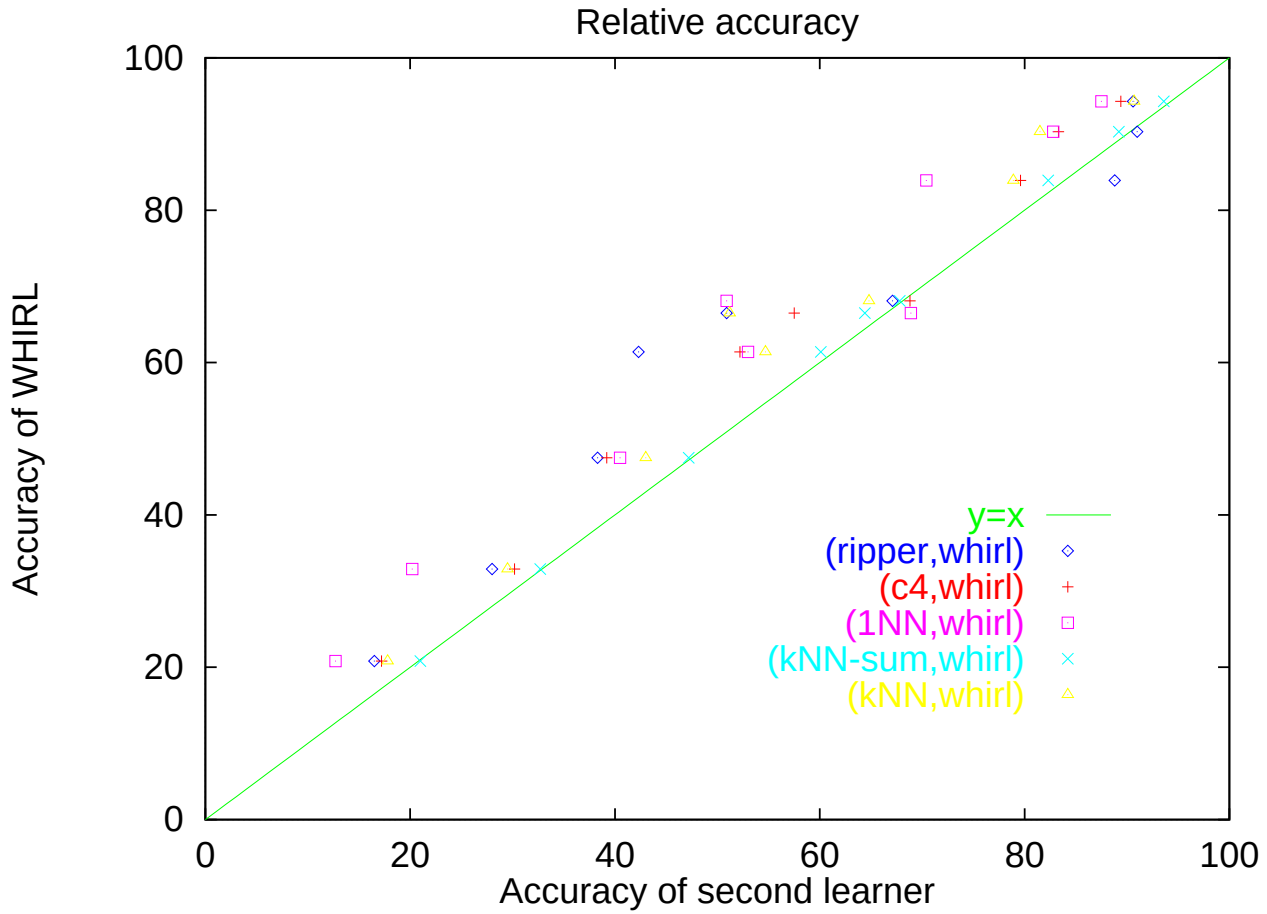
Using WHIRL for Classification

- Created nine representative datasets using data from Web.
- All instances were short "names"
  - *book title:* inst="The Humbugs of the World by P. T. Barnum (page images at MOA)", class="General Works"
  - *company name:* inst="National City Corporation", class="Banks–Midwest"
  - Also bird names, Web page titles, ...
- # classes ranged from 6 to 228, #instances ranged from ≈ 300 to ≈ 3000.

Benchmark classification problems

problem	#train/ #test	#classes/ #terms	text-valued field/label
memos	334/10cv	11/1014	document title/category
cdroms	798/10cv	6/1133	CDRom game name/category
birdcom	914/10cv	22/674	common name of bird/phylogenic order
birdsci	914/10cv	22/1738	common+sci name/phylogenic order
hcoarse	1875/600	126/2098	company name/industry (coarse grain)
hfine	1875/600	228/2098	company name/industry (fine grain)
books	3501/1800	63/7019	book title/subject heading
species	3119/1600	6/7231	animal name/phylum
netvet	3596/2000	14/5460	URL title/category

Using WHIRL for Classification



Joint work with Haym Hirsh

Classification with "side information"

Consider classification...

- Observation: Performance can often be improved by obtaining additional features about the entities involved.
- Question: Can performance be improved using weaker "side information"—like additional features that might or might not be about the entities involved in the classification task?

Instance		Label
Itzak Perlman	BMG	classic
Billy Joel	RCA	pop
Metallica	...	pop
...	...	...

Goal: from the data above, learn to classify musical artists as classical *vs.* popular.

Basic ideas: introduce new features for artist names that

- appear in certain lists or tables; (e.g., italicized names in the 'Guest Artist' page)
- are modified by certain words (e.g., "K$\emptyset\emptyset$L")

Guest Artists: Spring 2000

- Apr 9, *Itzak Perlman*
- May 3, *Yo Yo Ma*
- May 17, *The Guanari Quartet*
- ...

Biff's K$\emptyset\emptyset$L Band Links

- Nine Inch Nails (new!)
- Metallica!! Rockin'! Anyone know where can I find some MP3s?
- ...

...

The extraction algorithm

1. From HTML pages, create a table of (possible-name, position) pairs.
2. Soft-join with instance names to get (instance-name, position) pairs.

   Position is a new feature for the instance.

3. Can also create features from (possible-name, header-word) pairs.

```
html(head(...),
     body(
       h2(K∅∅L Band Links),
         table(
           tr(td(Metallica),
              td(Nine Inch Nails (new!))),
           tr(td(Barry Manilow),
     ...
```

Instances:

...

Metallica

Nine Inch Nails

Itzak Perlman

...

(“K∅∅L Band Links”, www.biff.com/html_body_h1)
(“Metallica”, www.biff.com/html_body_table_tr_td)
(“Nine Inch Nails (new!)”, www.biff.com/html_body_table_tr_td)
(“Barry Manilow”, www.biff.com/html_body_table_tr_td)

soft-join with instances and threshold

```
        h2(K∅∅L Band Links),
           table(
             tr(td(Metallica),
                td(Nine Inch Nails (new!))),
             tr(td(Barry Manilow),
      ...
```

(*instance-name*, *position*)
(“Metallica”, www.biff.com/html_body_table_tr_td)
(“Nine Inch Nails”, www.biff.com/html_body_table_tr_td)
(“Barry Manilow”, www.biff.com/html_body_table_tr_td)

Features from "header words"

```
        h2(K∅∅L Band Links),
        |  table(
        |    tr(td((Metallica)),
        |       td(Nine Inch Nails (new!)))),
      ...
```

(*instance-name*, *position*)

("Metallica", www.biff.com/html_body_table_tr_td) ...

(*instance-name*, *header-word*)

("Metallica", "K00L")

("Metallica", "Band")

("Metallica", "Links")

...

Benchmark problems

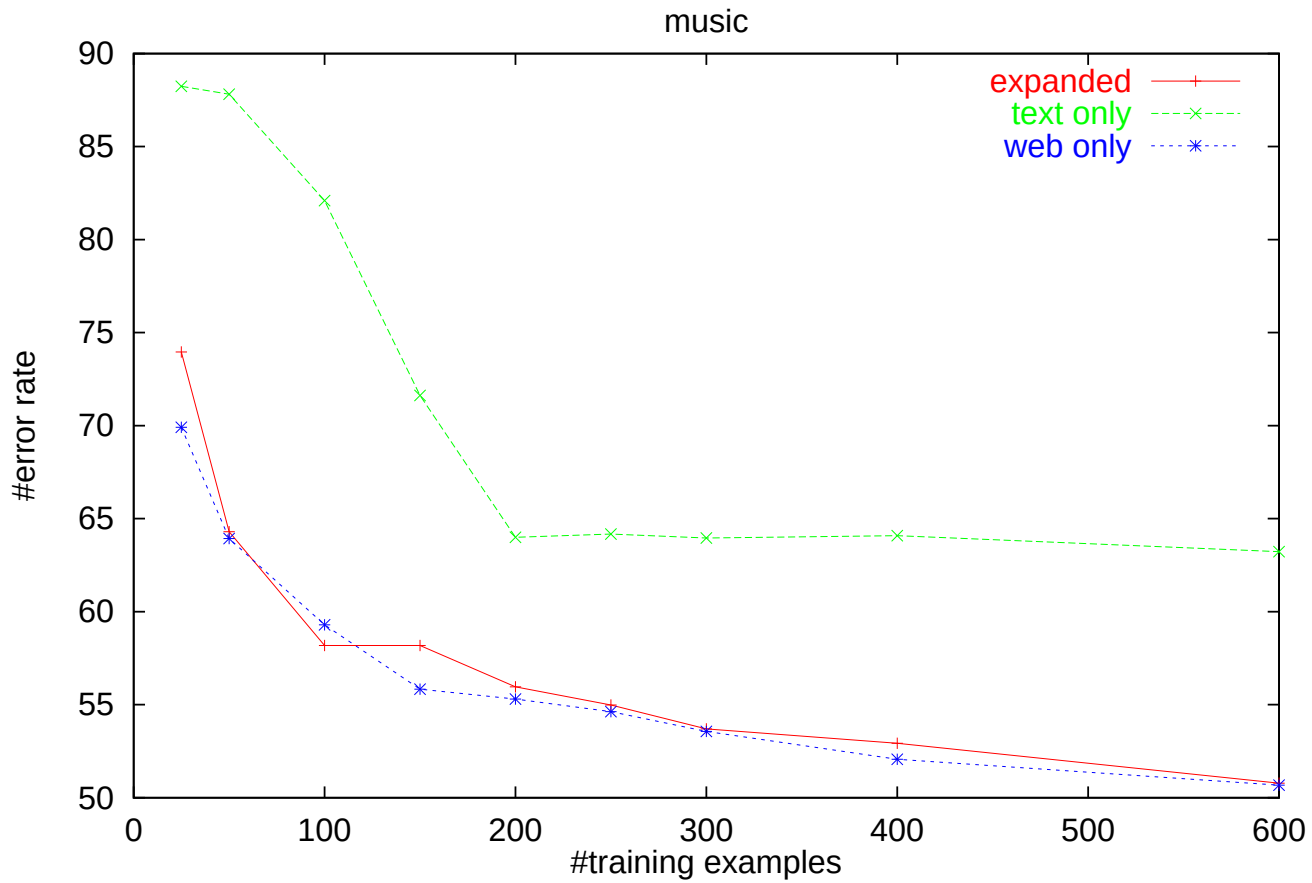
	#example	#class	#terms	#pages	#features added
music	1010	20	1600	217	1890
games	791	6	1133	177	1169
birdcom	915	22	674	83	918
birdsci	915	22	1738	83	533

- original data: names as bag-of-words
- music: (Cohen&Fan,WWW00) others: (Cohen&Hirsh,KDD98)
- note: test data must be processed as well (transduction).

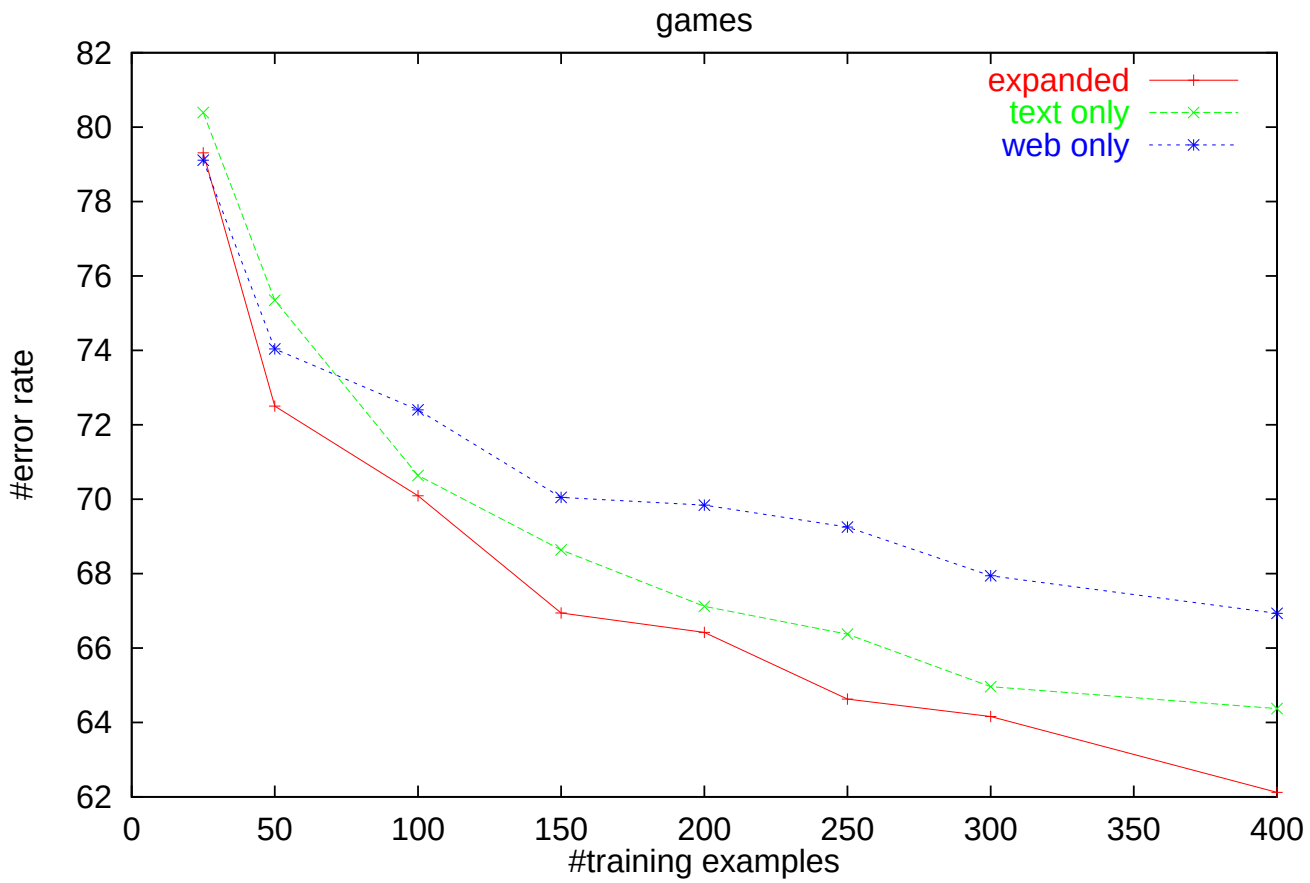
RIPPER: 200 training examples, 100 trials

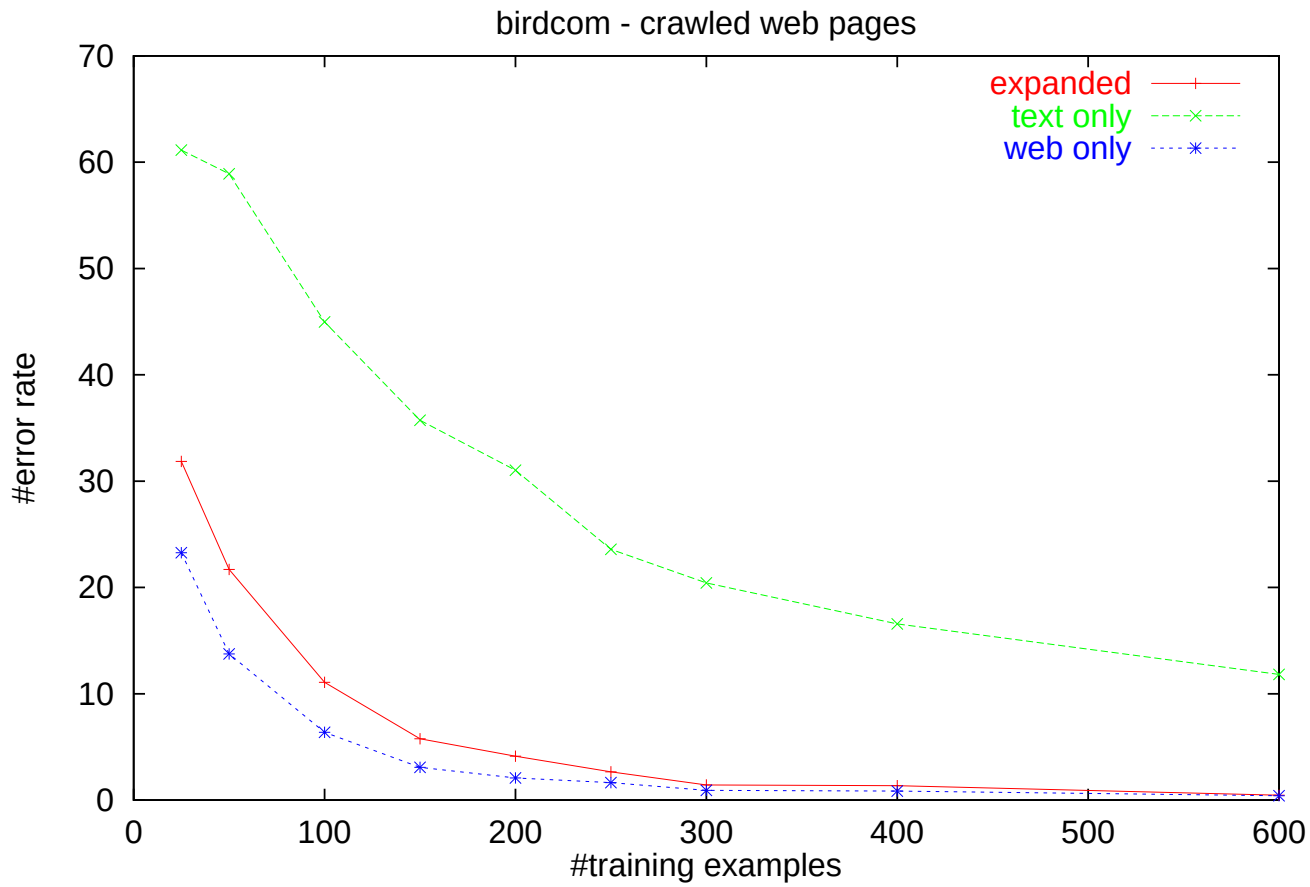
	W-L-T	Average error		Improvement
		text	expanded	
music	86-0-14	58.3	51.5	11.6%
cdroms	29-7-64	67.2	65.8	2.1%
birdcom	77-2-21	27.7	21.2	23.5%
birdsci	35-8-57	26.4	23.6	10.6%

Results (with RIPPER)



Results


games
#error rate
#training examples
expanded
text only
web only
82
80
78
76
74
72
70
68
66
64
62
0
50
100
150
200
250
300
350
400



Web pages automatically crawled—not sampled from WHIRL DB on birds.

The show so far:

- Motivation: why this is the big problem.
- WHIRL: Data model, query language, efficient implementation

Results & applications:

- Queries without formal identifiers
- Queries that don't require extraction
- Queries that generalize (Cohen & Hirsh, KDD98)
- Queries that automatically collect background knowledge for learning (Cohen ML2000, Cohen&Fan WWW2000)
- Comparison of TFIDF metric with other distance metrics for strings (Cohen, Ravikumar, Fienberg, in progress)

Other common distance metrics for strings

- **Bioinformatics:** edit distance metrics like Levenstein, Needleman-Wunch, Smith-Waterman, ...

  Can cope with misspellled tokens; not sensitive to frequency statistics (matching "Incorp" $\approx$ matching "Lucent").

- **Information retrieval:** token-based distance metrics like TFIDF (used in WHIRL), Jaccard, Dice, ..., statistical distances based on language modeling, ...

  Generally applied to long documents (prior to WHIRL).

- **Probabilistic record linkage:** statistical agreement measures like Fellegi-Sunter; *ad hoc* string distance metrics like Jaro, Jaro-Winkler.

  Generally used in a hand-constructed statistical model of matching/non-matching records, not as "hands-off" metrics.

Evaluation datasets

Name	Src	#Strings	#Tokens
animal	Whirl	5709	30,006
bird1	Whirl	377	1,977
bird2	Whirl	982	4,905
bird3	Whirl	38	188
bird4	Whirl	719	4,618
business	Whirl	2139	10,526
game	Whirl	911	5,060
park	Whirl	654	3,425
fodorZagrat	Ariadne	863	10,846
ucdFolks	Monge-Elkan	90	454
census	Winkler	841	5,765

Evaluation metrics

From IR community:

- 11-pt interpolated average precision, averaged across *all* datasets.
- Non-interpolated average precision, on *each* dataset.
- Maximum F1-measure on *each* dataset (see paper).

Edit distance metrics:

- Measure distance between strings s and t as cost of the least expensive sequence of edit operations that transform s to t.
- Example: to transform "Will Cohon" to "William Cohen" might use: copy, copy, copy, copy, insert(i), insert(a), insert(m), copy, copy, copy, copy, replace(o,e), copy.
- Different operations/costs lead to different metrics:
  - Levenstein: cost(cpy)=0, cost(ins(x))=1, cost(replace(x, y))=1.
- Minimal cost edit sequence usually can be found with dynamic programming in time $O(|s| \cdot |t|)$.

		W	i	l	l		C	o	h	o	n
	0	1	2	3	4	5	6	7	8	9
W	1	0	1	2	3	4	5	6	7	8
i	2	1	0	1	2	3	4	5	6	7
l	3	2	1	0	1	2	3	4	5	6
l	4	3	2	1	1	2	3	4	5	6
i	5	4	3	2	2	2	3	4	5	6
a	6	5	4	3	3	3	3	4	5	6
m	7	6	5	4	3	4	4	4	5	6
	8	7	6	5	4	3	4	5	5	6
C	9	8	7	6	5	4	3	4	5	6
o	10	9	8	7	6	5	4	3	4	5
h	11	10	9	8	7	6	5	4	4	5
e	12	11	10	9	8	7	6	5	5	4
n										

- Insert in s: move east, pay \$1
- Insert in t: move south, pay \$1
- Copy: move southeast, pay \$0
- Replace: move southeast, pay \$1
- Matrix i, j: cheapest path from northwest corner to i, j.
- Edit cost: cheapest path to southeast corner (4).

	t	e	d		c	h	e	n
n	**0**	**0**	**0**	0	0	0	0	1
e	**0**	**1**	**0**	**0**	0	0	1	0
d	**0**	**0**	**1**	**0**	**0**	0	0	0
	0	**0**	**0**	**1**	**0**	**0**	0	0
c	0	0	**0**	**0**	**1**	**0**	**0**	0
o	0	0	0	**0**	**0**	**0**	**0**	**0**
h	0	0	0	0	**0**	**1**	**0**	**0**
o	0	0	0	0	0	**0**	**0**	**0**
n	1	0	0	0	0	0	**0**	**1**

Jaro distance metric

$$\text{Jaro}(s,t) = \frac{1}{3}\cdot\left(\frac{|s'|}{|s|}+\frac{|t'|}{|t|}+\frac{|s'|-T_{s',t'}}{|s'|}\right)$$

- Find matching letters near the main diagonal, then find "common parts" of s and t: here $s' = t' =$"ed chn"
- Count transpositions in s' relative to t': $T_{s',t'}$
- Average fraction of s, t that are "common" with fraction of s' in the same order as t'.
- Jaro-Winkler: increase weight for weak matches if first few characters match well.

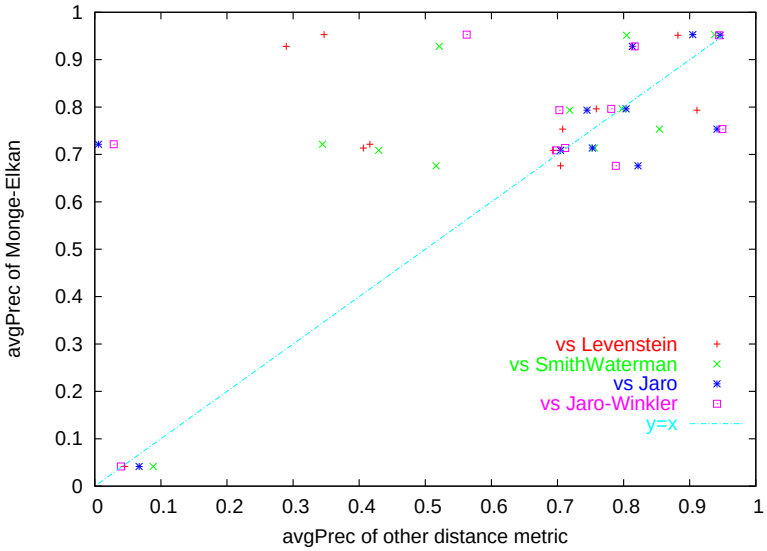
	t	e	d		c	h	e	n
n	**0**	**0**	**0**	0	0	0	0	1
e	**0**	**1**	**0**	**0**	0	0	1	0
d	**0**	**0**	**1**	**0**	**0**	0	0	0
	0	**0**	**0**	**1**	**0**	**0**	0	0
h	0	0	**0**	**0**	**0**	**1**	**0**	0
o	0	0	0	**0**	**0**	**0**	**0**	**0**
c	0	0	0	0	**1**	**0**	**0**	**0**
o	0	0	0	0	0	**0**	**0**	**0**
n	1	0	0	0	0	0	**0**	**1**

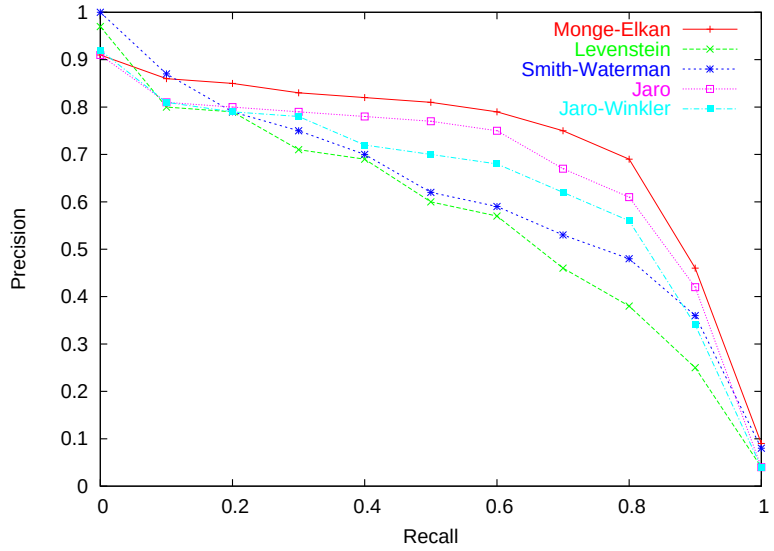
Jaro distance metric

$$\text{Jaro}(s,t) = \frac{1}{3}\cdot\left(\frac{|s'|}{|s|}+\frac{|t'|}{|t|}+\frac{|s'|-T_{s',t'}}{|s'|}\right)$$

- Find matching letters near the main diagonal, then find "common parts" of s and t: here s' ="ed hcn", t' ="ed chn"
- Count transpositions in s' relative to t': $T_{s',t'}$
- Average fraction of s, t that are "common" with fraction of s' in the same order as t'.
- Jaro-Winkler: increase weight for weak matches if first few characters match well.

Edit-distance and Jaro-based distances





Monge-Elkan: edit distance with well-tuned costs, affine gaps.

Token-based distance metrics

- View strings as sets (or bags) of tokens, S and T.
- Jaccard distance: $\frac{|S \cap T|}{|S \cup T|}$.
- View set S of tokens as a sample from an unknown distribution P_S, and consider differences between P_S and P_T:

$$\text{Jensen-Shannon}(S,T) = \frac{1}{2}\left(KL(P_S||Q) + KL(P_T||Q)\right)$$

where $KL(P||Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$, $Q = avg(P_S, P_T)$.

Token-based distance metrics

- Simplified Fellegi-Sunter: estimate log-odds of $P(S, T | s \text{ and } t \text{ match})$ as

$$\sum_{w \in S \cap T} \log \frac{1}{P(w)} - \sum_{w \in (S-T) \cup (T-S)} -k \log \frac{1}{P(w)}$$

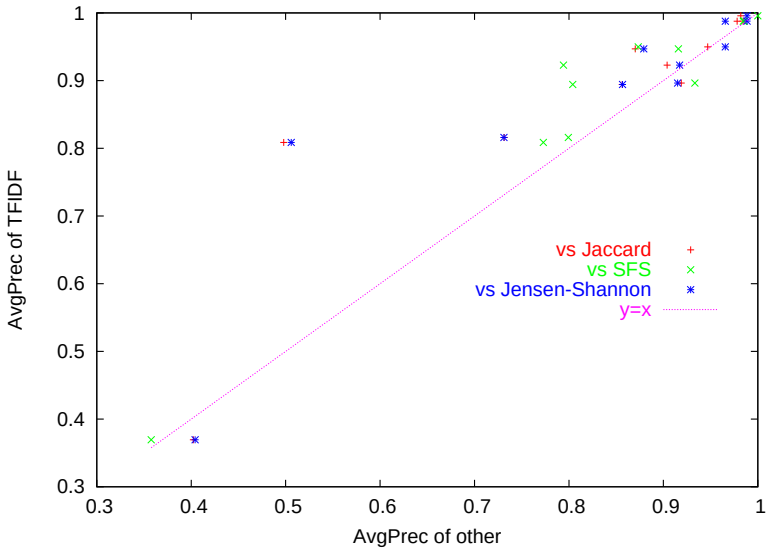
- TFIDF (WHIRL method): weight w by

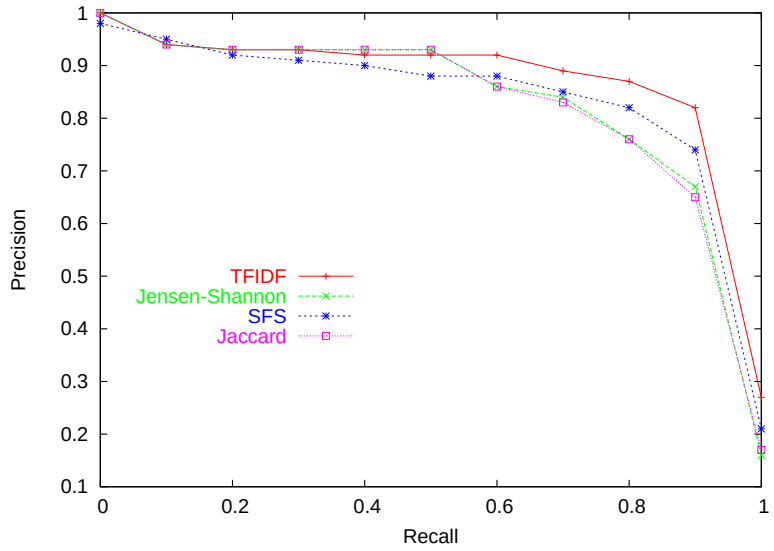
$$\log(1 + \text{freq of } w \text{ in string}) \times \log\left(\frac{\#\text{strings}}{\#\text{strings containing } w}\right)$$

Scale vectors to unit length, then similarity is:

$$\sum_{w \in S \cap T} weight(w, S) \cdot weight(w, T)$$

Token-based distance metrics





Hybrid distance measures

Assume sets of tokens S, T and a similarity measure for tokens $sim(w, v)$.

- Monge-Elkan propose a Level two similarity function between $S = \{w_1, \ldots, w_K\}$ and $T = \{v_1, \ldots, v_L\}$:

$$\text{Level2}(S, T) = \frac{1}{K} \sum_{i=1}^{K} \max_{j=1}^{L} sim(w_i, v_j)$$

Hybrid distance measures

- We propose a "softer" TFIDF measure. Recall:

$$\text{TFIDF}(S,T) = \sum_{w \in S \cap T} weight(w,S) \cdot weight(w,T)$$

$$\text{SoftTFIDF}(S,T) = \sum_{w \in CLOSE(\theta,S,T)} weight(w,S) \cdot weight(w,T) \cdot c(w,T)$$

Hybrid distance measures

- We propose a "softer" TFIDF measure:

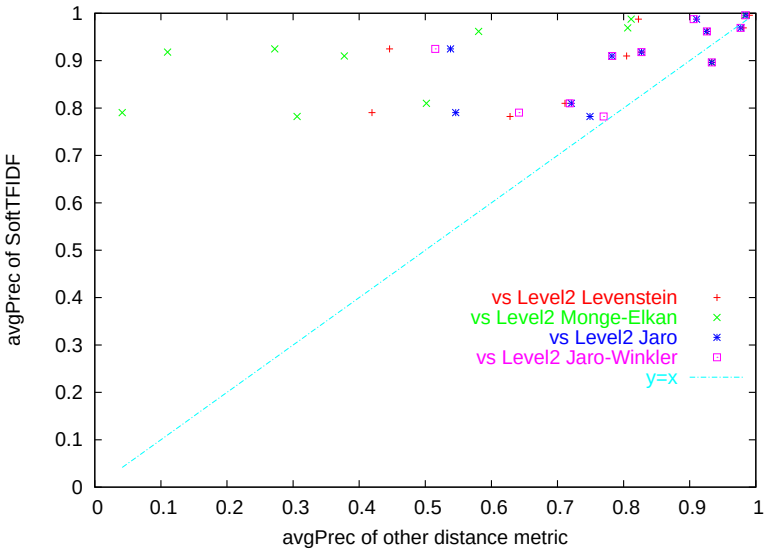
$$\text{SoftTFIDF}(S,T) = \sum_{w \in CLOSE(\theta,S,T)} weight(w,S) \cdot weight(w,T) \cdot c(w,T)$$

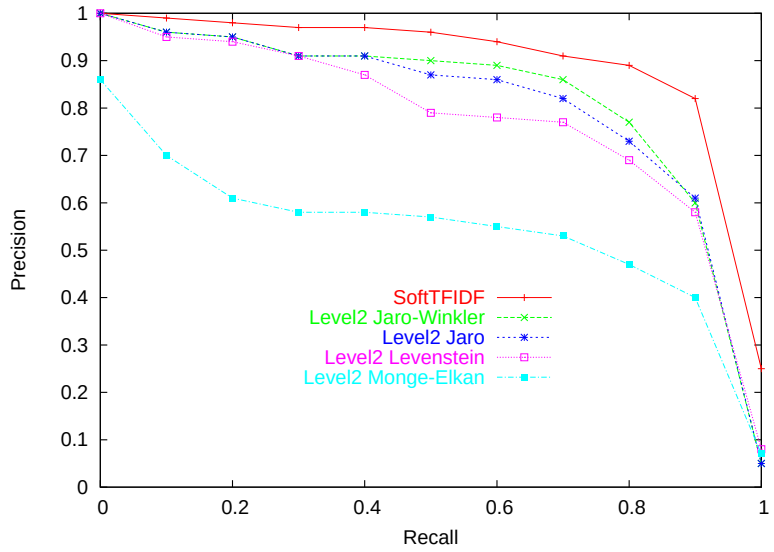
  where

  - $CLOSE(\theta, S, T) = \{w \in S : \exists v \in T \text{ and } sim(w,v) > \theta\}$
    (Similar tokens in S and T)
  - $c(w,T) = \max_{v \in T} sim(w,v)$
    (Similarity to closest token in T)

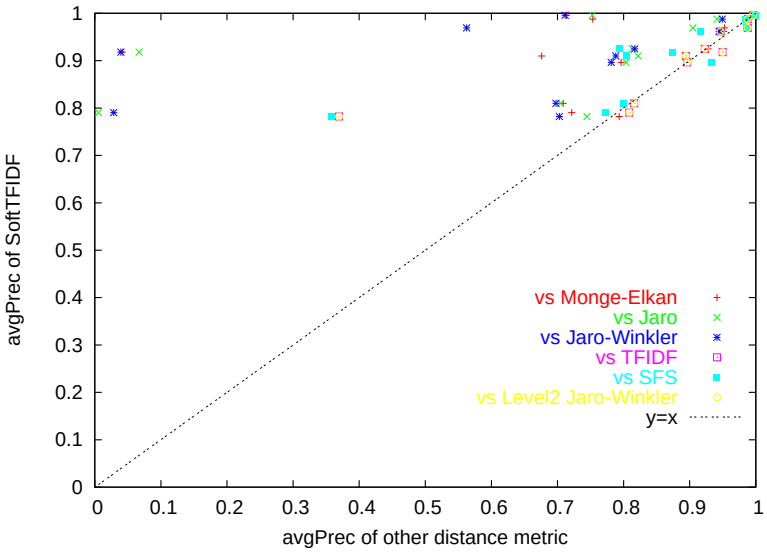
- Will use $\theta = 0.9$, *sim*=Jaro-Winkler.

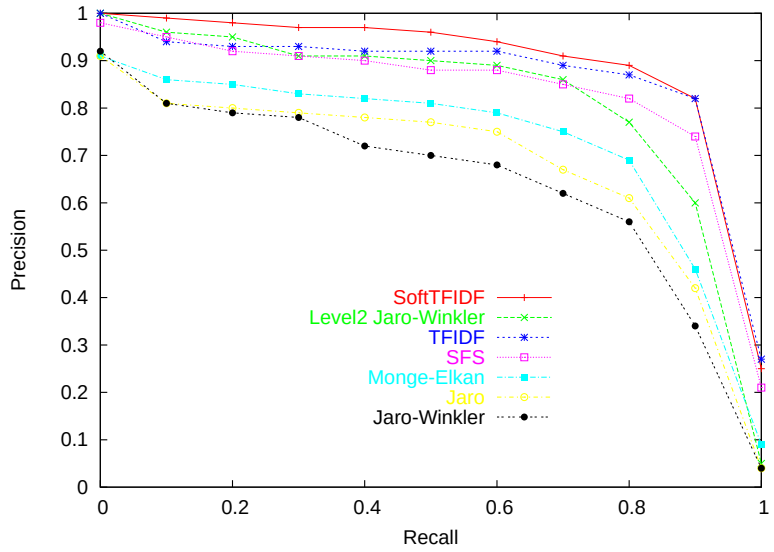
Hybrid distance metrics





Grand summary of best metrics





Prospective test: two more datasets

	UVA (Monge-Elkan)		CoraATDV (JPRC)	
Method	MaxF1	AvgPrec	MaxF1	AvgPrec
SoftTFIDF	0.89	0.91	0.85	0.914
TFIDF	0.79	0.84	0.84	0.907
SFS	0.71	0.75	0.82	0.864
Level2 J-W	0.73	0.69	0.76	0.804

Conclusions

- The next step (?) after distributing text world-wide: learn how to reason with a world-wide knowledge base.
- Integrating structured data from multiple sources is a crucial problem.
  - Object identity issues dominate in many domains.
- WHIRL efficiently propogates uncertainty about object identity.
- TFIDF distance is fast and surprisingly robust.
- WHIRL data model and query language allow an intermediate between "document delivery" and "deductive" information systems.

Beyond data integration, WHIRL is useful for many other tasks:

- Querying imperfectly extracted data
- Queries that generalize (Cohen & Hirsh, KDD98)
- Automatically collecting features for learning (Cohen, ML2000)
- Queries that suggest extraction rules (Cohen, AAAI99)
- Content-based recommendation (Basu et al, JAIR2001)
- Bootstrapping-based extraction of relations from text (Agichtein & Gravano, DL2000)
- Extensions for semistructured data (Chinenyanga & Kushmerick, SIGIR2001)
- Stochastic matrix multiplication for better performance on conjuctive chain queries (Gravano et al, WWW2003)