Chapter 8: Information Extraction (IE)

8.6 Knowledge Acquistion

Goal:
find **all instances** of a given (unary, binary, or N-ary) relation (or a given set of such relations) in a **large corpus** (Web, Wikipedia, newspaper archive, etc.)

Example targets:

Cities(.), Rivers(.), Countries(.), Movies(.), Actors(.), Singers(.), Headquarters(Company,City), Musicians(Person, Instrument), Synonyms(.,.), ProteinSynonyms(.,.), ISA(.,.), IsInstanceOf(.,.), SportsEvents(Name,City,Date), etc.

Assumption:
There is an NER tagger for each individual entity class (e.g. based on:
PoS tagging + dictionary-based filtering + window-based classifier or rule-based pattern matcher)

Online demos: http://dewild.cs.ualberta.ca/
http://www.cs.washington.edu/research/knowitall/

Simple Pattern-based Extraction (Staab et al.)

0) define phrase patterns for relation of interest (e.g. IsInstanceOf)

1) extract proper nouns (e.g. the Blue Nile)

2) for each document

   use proper nouns in doc and phrase patterns

   to generate candidate phrases

   (e.g. rivers like the Blue Nile, the Blue Nile is a river, life is a river)

3) query large corpus (e.g. via Google)

   to estimate frequency of (confidence in) candidate phrases

4) for each candidate instance of relation

   combine frequencies (confidences) from different phrases

   e.g. by summation or weighted summation with weights learned from training corpus

5) define threshold for selecting instances

Phrase Patterns for IsInstanceOf

Hearst patterns (M. Hearst 1992):

H1: CONCEPTs such as INSTANCE

H2: such CONCEPT as INSTANCE

H3: CONCEPTs, (especially | including) INSTANCE

H4: INSTANCE (and | or) other CONCEPTs

Definites patterns:

D1: the INSTANCE CONCEPT

D2: the CONCEPT INSTANCE

Apposition and copula patterns:

A: INSTANCE, a CONCEPT

C: INSTANCE is a CONCEPT

Unfortunately, this approach does not seem to be robust

Example Results for Extraction based on Simple Phrase Patterns

INSTANCE	CONCEPT	frequency
Atlantic	city	1520837
Bahamas	island	649166
USA	country	582775
Connecticut	state	302814
Caribbean	sea	227279
Mediterranean	sea	212284
South Africa	town	178146
Canada	country	176783
Guatemala	city	174439
Africa	region	131063
Australia	country	128067
France	country	125863
Germany	country	124421
Easter	island	96585
St. Lawrence	river	65095
Commonwealth	state	49692
New Zealand	island	40711
St. John	church	34021
EU	country	28035
UNESCO	organization	27739
Austria	group	24266
Greece	island	23021

Source:
Cimiano/Handschuh/Staab:
WWW 2004

SNOWBALL: Bootstrapped Pattern-based Extraction (Agichtein et al.)

Key idea (see also S. Brin: WebDB 1998):

start with small set of **seed tuples** for relation of interest

find **patterns** for these tuples, assess confidence, select best patterns

repeat

  find **new tuples** by matching patterns in docs

  find **new patterns** for tuples, assess confidence, select best patterns

Example:

seed tuples for Headquarters (Company, Location):
{(Microsoft, Redmond), (Boeing, Seattle), (Intel, Santa Clara)}
patterns: LOCATION-based COMPANY, COMPANY based in LOCATION
new tuples:
{(IBM Germany, Sindelfingen), (IBM, Böblingen), ...}
new patterns:
LOCATION is the home of COMPANY, COMPANY has a lab in LOCATION, ...

SNOWBALL Methods in More Detail (1)

Vector-space representation of patterns (SNOWBALL-VSM):

pattern is **5-tuple (left, X, middle, Y, right)**
where left, middle, right are term vectors with term weights

Algorithm for adding patterns:

find **new tuple (x,y)** in corpus & construct **5-tuple around (x,y);**
if **cosine sim** against 5-tuples of known pattern > sim-threshold then
  add 5-tuple around (x,y) to set of **candidate patterns**;
cluster candidate patterns;
use **cluster centroids** as new patterns;

Algorithm for adding tuples:

if **new tuple t** found by pattern P **agrees with known tuple**
then P.pos++ else P.neg++;
confidence(P) := P.pos / (P.pos + P.neg);
confidence(tuple t) := $1 - \prod_{P \in patterns} (1 - confidence(P) \cdot sim(t, P))$
if confidence(t) > conf-threshold then add t to relation

SNOWBALL Methods in More Detail (2)

VSM representation fails in situations such as:
... where Microsoft is located whereas the Silicon Valley startup ...

<u>Sequence representation of patterns (SNOWBALL-MST):</u>
pattern is term sequence with don‘t-care terms
Example: ... near Boeing‘s renovated Seattle headquarters ...
$\rightarrow$ near X ‘s * Y headquarters

<u>Algorithm:</u>
use Sparse Markov Transducer (related to HMMs) to estimate
confidence(t) := P[t | pattern sequence]

SNOWBALL Combination Methods

combine SNOWBALL-VSM and SNOWBALL-MST
(and other methods ...) by

- intersections/unions of patterns and/or new tuples
- weighted mixtures of patterns and/or tuples
- voting-based ensemble learning
- co-training

etc.

Evaluation

Ground truth:
either
• hand-extract all instances from small test corpus
or
• retrieve all instances from larger corpus
that occur in an ideal result derived from a collection of explicit facts
(e.g. CIA factbook and other almanachs)

then use IR measures:
• precision
• recall
• F1

Evaluation of SNOWBALL Methods

finding Headquarters instances in 142000 newspaper articles
with ground truth = newspaper corpus ∩ Hoover's online

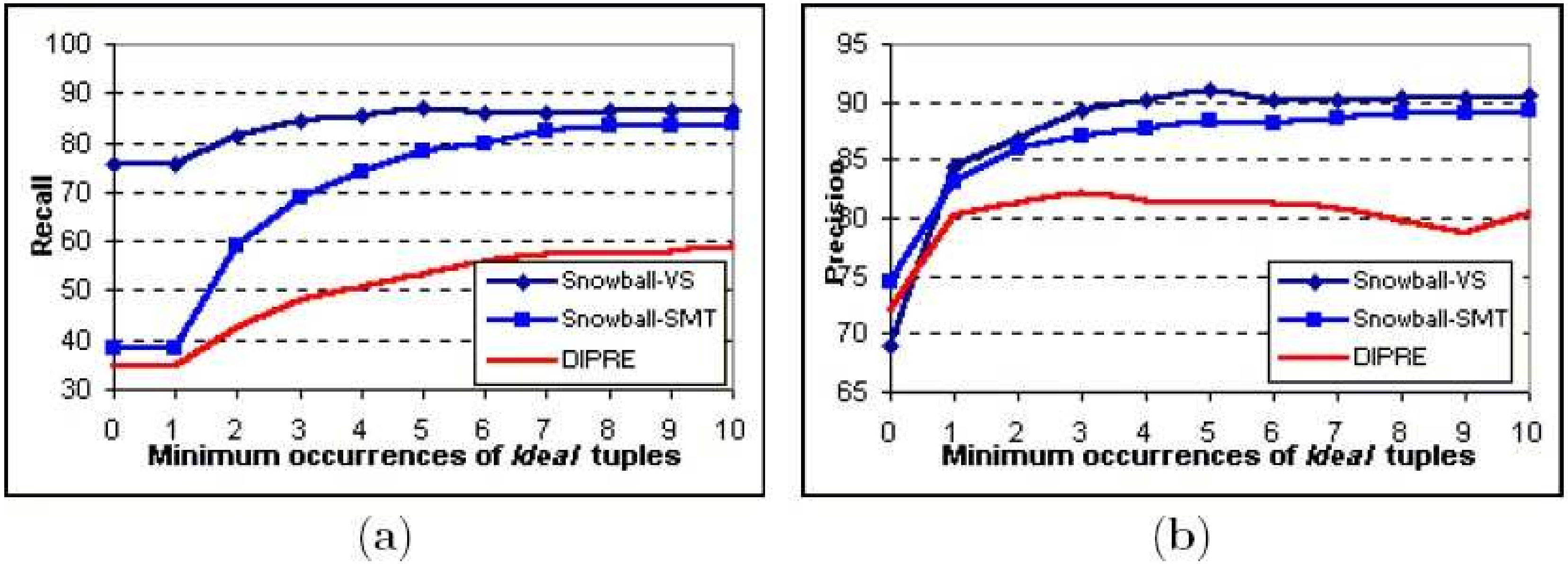


Figure 4: Recall (a) and precision (b) of DIPRE, *Snowball-VS*, and *Snowball-SMT* (test collection).

with parameter settings fit based on training collection (36000 docs)

QXtract: Quickly Finding Useful Documents

In very large corpus, scanning all docs by SNOWBALL
may be too expensive
→ find and process only potentially useful docs

Method:
sample := randomly selected docs ∪ **query-result (seed-tuples terms);**
run SNOWBALL on sample;
UsefulDocs := docs in sample that contain relation instance
UselessDocs := sample – UsefulDocs;

run feature-selection techniques or classifier
to identify most **discriminative terms**
between UsefuDocs and UselessDocs (e.g. MI, BM25 weights, etc.);
generate queries with small number of best terms from UsefulDocs;

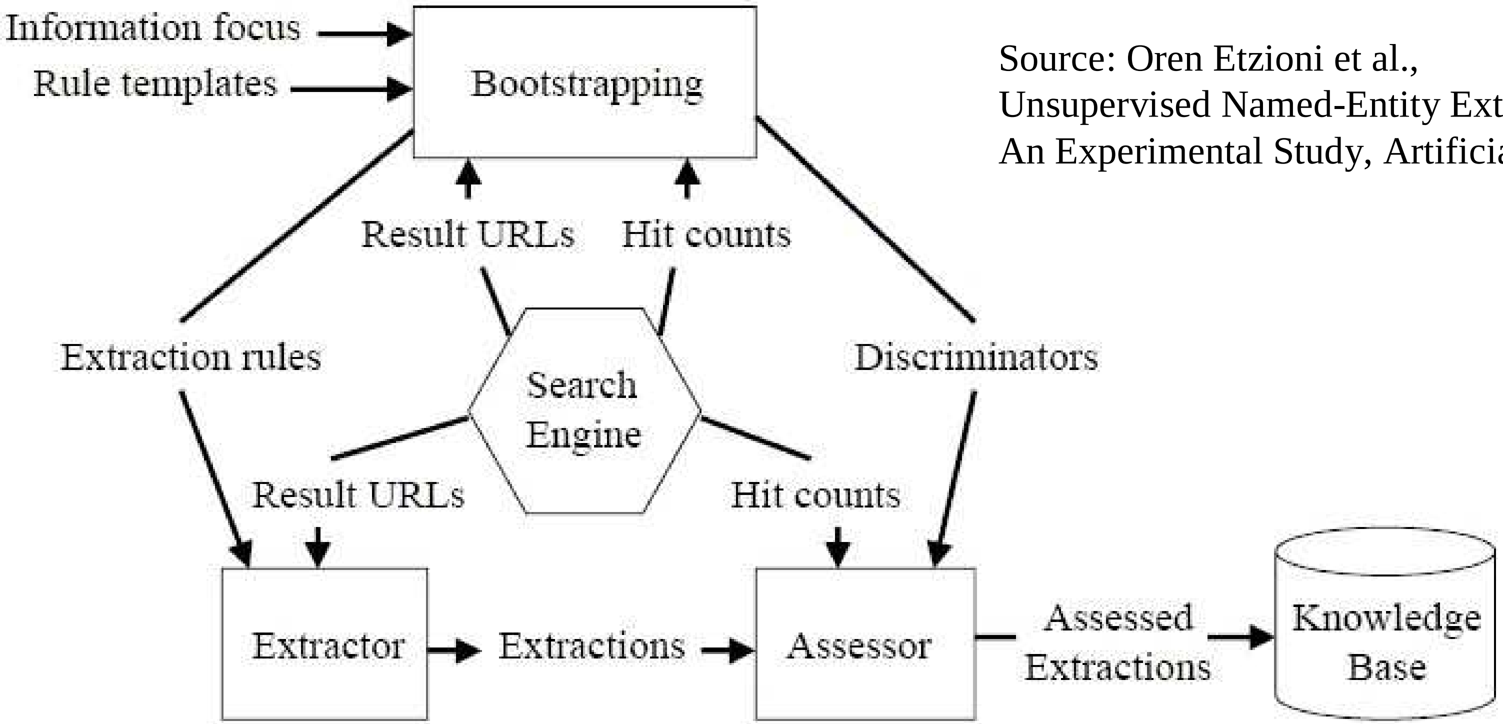
KnowItAll: Large-scale, Robust Knowledge Acquisition from the Web

Goal: find all instances of relations
such as cities(.), capitalOf(city, country), starsIn(actor, film), etc.

- Almost-Unsupervised **Extractor** with **Bootstrapping**:
  - Start with general patterns (e.g.: X such as Y)
  - Learn domain-specific patterns
    (e.g.: towns such as Y, cities such as Y)
  - Extended pattern learning
- **Assessor** evaluates quality of extracted instances
  and learned patterns
- Alternate between Extractor and Assessor

Collections and demos: http://www.cs.washington.edu/research/knowitall/
(emphasis on unary relations: instances of object classes)

KnowItAll Architecture



Source: Oren Etzioni et al.,
Unsupervised Named-Entity Extraction from the Web:
An Experimental Study, Artificial Intelligence 2005

<u>Bootstrap:</u>

create rules R, queries Q,
discriminators D
repeat
Extractor (R, Q) finds facts E
Assessor (E, D) adds facts to KB
until Q is exhausted or #facts > n

<u>Extractor:</u>

Select queries from Q and send to SE
for each returned web page w do
Extract fact e from w using rule for query q

<u>Assessor:</u>

for each fact e in E do
assign prob. p to e using NB class. based on D
add e, p to KB

KnowItAll Extraction Rules

Generic pattern (rule template)

Predicate: Class1
Pattern: NP1 „such as“ NPList2
Contraints: head(NP1) = plural(label(Class1)) &
properNoun(head(each(NPList2)))
Bindings: Class1(head(each(NPList2)))

Domain-specific pattern

Predicate: City
Label: City
Keywords: „cities such as“, „urban centers“
Pattern: NP1 „such as“ NPList2
Contraints: head(NP1) = „cities“ &
properNoun(head(each(NPList2)))
Bindings: City(head(each(NPList2)))

Domain-specific pattern for binary relation

Predicate: CEOofCompany (Person, Company)
...
Pattern: NP1 „ , “ P2 NP3
Contraints: properNoun(NP1) & P2 = „CEO of“
& properNoun(NP3)
Bindings: CEOofCompany (NP1, NP3)

8 generic patterns for unary,
2 example patterns for binary

NP “and other” <class1>
NP “or other” <class1>
<class1> “especially” NPList
<class1> “including” NPList
<class1> “such as” NPList
“such” <class1> “as” NPList
NP “is a” <class1>
NP “is the” <class1>

<class1> “is the” <relation> <class2>
<class1> “,” <relation> <class2>

NP analysis crucial, *e.g.*
head(NP) is last noun:
China is a country in Asia
vs.
Garth Brooks is a country singer

KnowItAll Bootstrapping

Automatically creating domain-specific
extraction rules, queries, and discriminator phrases

1) Start with class/relation name and keywords
   e.g. for unary MovieActor: movie actor, actor, movie star
   e.g. for binary capitalOf: capital of, city, town, country, nation
2) Substitute names/keywords and characteristic phrases for variables in generic rules *(e.g. X such as Y)* to generate
   - **new extraction rules** *(e.g. cities such as Y, towns such as Y),*
   - **queries for retrieval** *(e.g. cities, towns, capital),* and
   - **discriminators for assessment** *(e.g. cities such as)*
3) Repeat with extracted facts/sentences

Extraction rules aim to increase coverage,
Discriminators aim to increase accuracy

KnowItAll Assessor

<u>Input:</u>

- Extracted fact e (relation instance)

  e.g.: City(Paris)

- Discriminator phrases D (automatically generated from class name, ≥ 2 keywords of rules, learned extended patterns)

  e.g.: „X is a city“, „X and other towns“, „X is the capital of“, etc. [X→Paris]

<u>Output:</u>

- Confidence in (probability of) validity of e

Compute by queries to SE:
pointwise mutual information $PMI(e,d) = \frac{|Hits(e \cup d)|}{|Hits(e)|}$

PMI scores for e form feature vector for e
fed into Naive Bayes classifier for validity of e

NBC for relation E trained by
positive discriminators for E with highest PMI scores
and pos. discr. for other relations as negative discr. for E

Queries are scalability bottleneck
→ probabilistic model for estimation

KnowItAll Example

interested in Cities (.), States (.), Countries (.), …

Bootstrapping finds facts E:

Cities(London), Cities(Rome), Cities(Dagupan), Cities(Shakhrisabz), …

States(Oregon), States(Arizona), States(Georgia), …

and discriminators D (with PMI scores):

„X is a city“, „X and other towns“, „cities X“, „cities such as X“, „cities including X“

Generate query ***„and other cities“*** from rule: ***NP „and other cities“,***

and retrieve:

„Short flights connect Casablance with Fes and other cities.“

„The ensemble has performed concerts throughout the East Coast and other cities.“

Extractor extracts candidates e: ***Cities(Fes), Cities(East Coast)***

Assessor submits 6 queries for each e:

„Fes“, „Fes is a city“, „Fes and other towns“, etc.

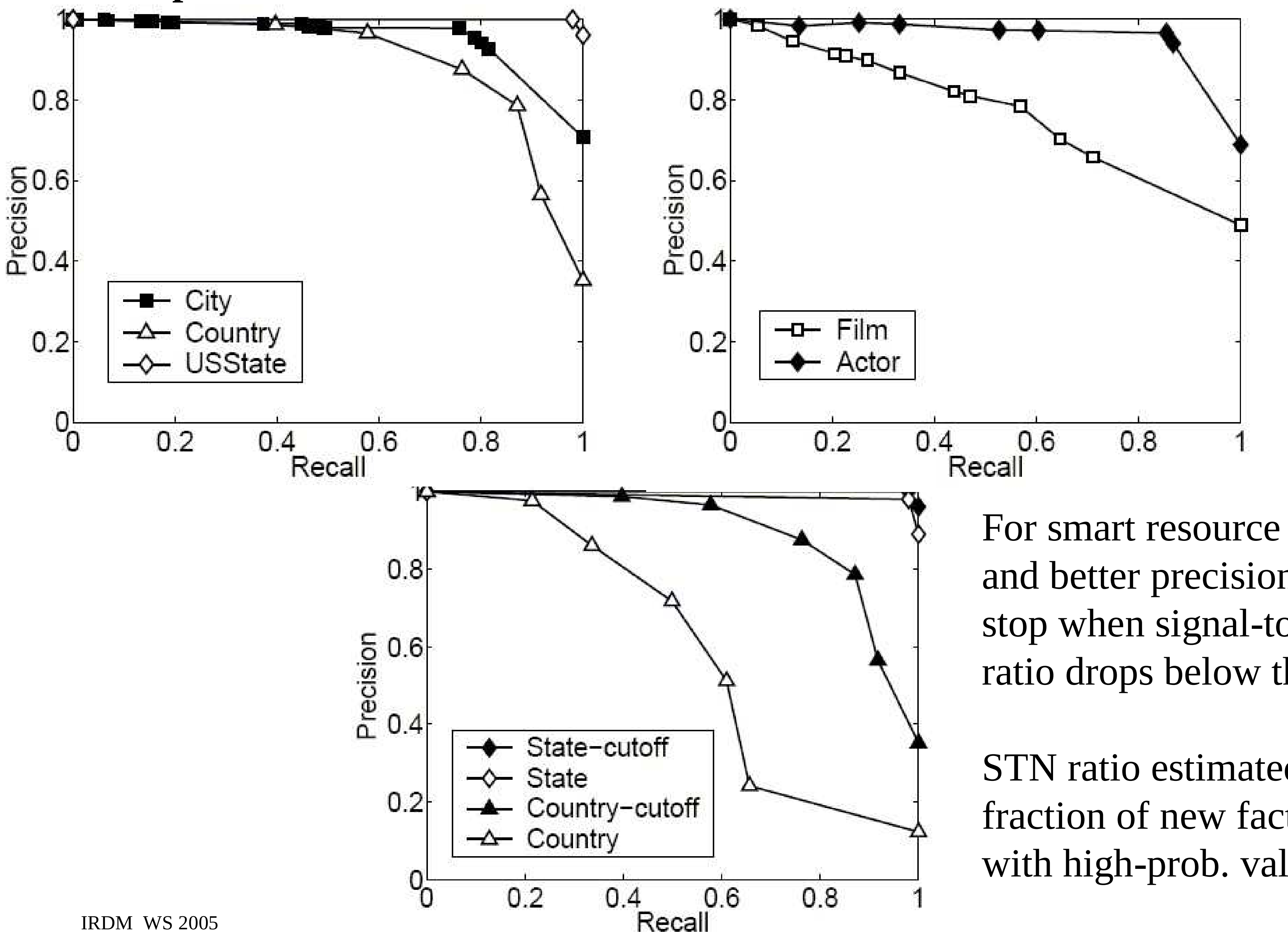
„East Coast“, „East Coast is a city“, „East Coast and other towns“, etc.

It computes PMI scores and uses NBC to test validity of each e

$\rightarrow$ ***accept Cities(Fes), reject Cities(East Coast)***

KnowItAll Experiments

with Tipster Gazetteer and IMDB as ground truth



For smart resource usage
and better precision
stop when signal-to-noise
ratio drops below threshold

STN ratio estimated by
fraction of new facts
with high-prob. validity

KnowItAll Extensions

Learning additional extraction patterns:

- Consider LR-rule-style extractors around extracted fact
  (***e.g. headquartered in X, mayor of X is <person>)***
- Assess their precision/recall by statistics from previous extractions
  (new rules can serve as extractors and/or discriminators)

Subclass handling:

- Identify candidates for ISA (hypernymy) relation,
  get statistics on instances, check WordNet, etc.
  (e.g. capital $\subseteq$ city, stem cell researcher $\subseteq$ microbiologist $\subseteq$ biologist $\subseteq$ scientist)
- Improve recall by having the Extractor consider all subclasses together

List extraction:

- Improve recall by retrieving HTML lists (<table>) and
  assessing their entries (<td>) based on previous extractions
  (cf. Google sets: http://labs.google.com/sets)

Additional Literature for Chapter 8

<u>IE Overview Material:</u>

- S. Chakrabarti, Section 9.1: Information Extraction
- N. Kushmerick, B. Thomas: Adaptive Information Extraction: Core Technologies for Information Agents, AgentLink 2003
- H. Cunningham: Information Extraction, Automatic, to appear in: Encyclopedia of Language and Linguistics, 2005, http://www.gate.ac.uk/ie/
- W.W. Cohen: Information Extraction and Integration: an Overview, Tutorial Slides, http://www.cs.cmu.edu/~wcohen/ie-survey.ppt
- S. Sarawagi: Automation in Information Extraction and Data Integration, Tutorial Slides, VLDB 2002, http://www.it.iitb.ac.in/~sunita/

Additional Literature for Chapter 8

Rule- and Pattern-based IE:

- M.E. Califf, R.J. Mooney: Relational Learning of Pattern-Match Rules for Information Extraction, AAAI Conf. 1999
- S. Soderland: Learning Information Extraction Rules fro Semi-Structured and Free Text, Machine Learning 34, 1999
- Arnaud Sahuguet, Fabien Azavant: Looking at the Web through XML Glasses, CoopIS Conf. 1999
- V. Crescenzi, G. Mecca: Automatic Information Extraction from
- Large Websites, JACM 51(5), 2004
- G. Gottlob, C. Koch, R. Baumgartner, M. Herzog, S. Flesca: The Lixto Data Extraction Project, PODS 2004
- A. Arasu, H. Garcia-Molina: Extracting Structured Data from Web Pages, SIGMOD 2003
- A. Finn, N. Kushmerick: Multi-level Boundary Classification for Information Extraction, ECML 2004

Additional Literature for Chapter 8

HMMs and HMM-based IE:

- Manning / Schütze, Chapter 9: Markov Models
- Duda/Hart/Stork, Section 3.10: Hidden Markov Models
- W.W. Cohen, S. Sarawagi: Exploiting dictionaries in named entity extraction: combining semi-Markov extraction processes and data integration methods, KDD 2004

Entity Rconciliation:

- W.W. Cohen: An Overview of Information Integration, Keynote Slides, WebDB 2005, http://www.cs.cmu.edu/~wcohen/webdb-talk.ppt
- S. Chaudhuri, R. Motwani, V. Ganti: Robust Identification of Fuzzy Duplicates, ICDE 2005

Knowledge Acquisition:

- O. Etzioni: Unsupervised Named-Entity Extraction from the Web: An Experimental Study, Artificial Intelligence 165(1), 2005
- E. Agichtein, L. Gravano: Snowball: extracting relations from large plain-text collections, ICDL Conf., 2000
- E. Agichtein, V. Ganti: Mining reference tables for automatic text segmentation, KDD 2004
- IEEE CS Data Engineering Bulletin 28(4), Dec. 2005, Special Issue on Searching and Mining Literature Digital Libraries