Information Extraction and Weakly-supervised Learning

19th European Summer School in Logic, Language and Information
13th – 17th August 2007

Course Overview

- Examine one language processing technology (Information Extraction) in depth
- Focus on machine learning approaches
  - Particularly semi-supervised algorithms

Schedule

1. Introduction to Information Extraction
2. Relation Identification (1)
   Learning patterns: supervised \( \rightarrow \) weakly supervised
3. Relation Identification (2)
   Counter training; WordNet-based approach
4. Named entity extraction
   Terminology recognition
5. Information Extraction Pattern Models
   Comparison of four alternative models

Course Home Page


- Materials, links

Part 1:
Introduction to Information Extraction
Overview

- Introduction to Information Extraction (IE)
  - The IE problem
  - Applications
  - Approaches to IE
- Evaluation in IE
  - The Message Understanding Conferences
  - Performance measures

What is Information Extraction?

- Huge amounts of knowledge are stored in textual format
- Information Extraction (IE) is the identification of specific items of information in text
- These can be used to fill databases, which can be queried later

Example

October 14, 2002, 4:00 a.m. PT
For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a “cancer” that stifled technological innovation.

“We can be open source. We love the concept of shared source,” said Bill Veghte, a Microsoft VP.

“That’s a super-important shift for us in terms of code access.”

Richard Stallman, founder of the Free Software Foundation, countered saying...

<table>
<thead>
<tr>
<th>NAME</th>
<th>TITLE</th>
<th>ORGANIZATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill Gates</td>
<td>CEO</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Bill Veghte</td>
<td>VP</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Richard Stallman</td>
<td>founder</td>
<td>Free Software Foundation</td>
</tr>
</tbody>
</table>

Applications

- Many applications for IE:
  - Competitive intelligence
  - Drug discovery
  - Protein–protein interactions
  - Intelligence (e.g., extraction of information from emails, telephone transcripts)

IE Process

- Information Extraction is normally carried out in a two-stage process:
  1. Name identification
  2. Event extraction
**Name Identification and Classification**

- First stage in majority of IE systems is to identify the named entities in the text.
- The names in text will vary according to the type of text:
  - Newspaper texts will contain the names of people, places and organisations.
  - Biochemistry articles will contain the names of genes and proteins.

**News Example**

"Capt. Andrew Ahab was appointed vice president of the Great White Whale Company of Salem, Massachusetts."

**Biomedical Example**

"Localization of SpoIIE was shown to be dependent on the essential cell division protein FtsZ."

**Event Extraction**

- Event extraction is often carried out after named entity identification.
- The aim is to identify all instances of a particular relationship or event in text.
- A template is used to define the items which are to be extracted from the text.

**News Example**

"Neil Marshall, vice president of Ford Motor Corp., has been appointed president of DaimlerChryslerToyota."

**Biomedical example**

"Localization of SpoIIE was shown to be dependent on the essential cell division protein FtsZ."

In this case the “event” is an interaction between a gene and protein.
**Approaches to Building IE Systems**

1. Knowledge Engineering Approaches
   - Information extracted using patterns which match text
   - Patterns written by human experts using their own knowledge of language and of the subject domain (by analysing text)
   - Very time consuming
2. Learning Approaches
   - Learn rules from text
   - Can require large amounts of annotated text

**Supervised and Unsupervised Learning**

- Machine learning algorithms can be divided into two main types:
  - **Supervised**: algorithm is given examples of text marked (annotated) with what should be learned from it (e.g., named entities or events)
  - **Unsupervised** (or weakly supervised) algorithm is given a large amount of raw text (and a few examples)

**Constructing Event Recognisers**

- Create regular-expression patterns which
  - match text, and
  - contain instructions for filling templates

- Knowledge engineering: write patterns manually
- Learning: infer patterns from text

**Analysing Sentence Structure**

- One way to analyse the sentence in more detail is to analyse its structure
- This process is known as parsing
- One example of how this could be used is to identify groups of related words

**IE is difficult as the same information can be expressed in a wide variety of ways**

1. IBM has appointed Neil Marshall as president.
2. IBM announced the appointment of Neil Marshall as president.
3. IBM declared a special dividend payment and appointed Neil Marshall as president.
5. IBM has made a major management shuffle. The company appointed Neil Marshall as president.
Information Extraction and Weakly-supervised Learning

Example

- Sentence
  Ford has appointed Neil Marshall, 45, as president.

- Name identification
  Ford has appointed Neil Marshall, 45, as president.
  "Ford" Name type= organisation
  "Neil Marshall" Name type = person

- Noun Phrase analysis
  Ford has appointed Neil Marshall, 45, as president.
  "Ford" NP-head=organisation
  "Neil Marshall, 45," NP-head=person

- Verb Phrase analysis
  Ford has appointed Neil Marshall, 45, as president.
  "Ford" NP-head=organisation
  "Neil Marshall, 45," NP-head=person
  "has appointed" VP-head=appoint

- Event Extraction
  Person= "Neil Marshall"
  Company= "Ford"
  Position= "president"
  Start/leave= start

Dependency Analysis

- Dependency analysis of a sentence relate each word to other words which depend on it.
- Dependency analysis is popular as a computational model since relationships between words are useful
  - "The old dog" → "the" and "old" depend on "dog"
  - "John loves Mary" → "John" and "Mary" depend on "loves"

Example

"IBM named Smith, 54, as president"

```
  subject
   IBM

  object
   Smith

  mod
   54

  copredicate
   as

  pcomp
   named
```

- Dependencies labelled in this example

Example

The man on the hill
has the telescope

```
  saw
   John

  man
   the
   on
   with

  hill
   with

  telescope
```

- The man on the hill has the telescope
Dependency Parsers

- Dependency analysis for sentences can be automatically generated using dependency parsers

- Connexor Parser: http://www.connexor.com/demo/syntax/


Evaluation

- Information Extraction usually evaluated by comparing the performance of a system against a human judgement of the same text

- The events identified by the human are the gold standard

- IE evaluations started with the Message Understanding Conferences (MUCs), sponsored by the US government

MUC Conferences

  - Messages about naval operations

- MUC-3 (1991) and MUC-4 (1992)
  - News articles about terrorist activity

- MUC-5 (1993)
  - News articles about joint ventures and microelectronics

- MUC-6 (1995)
  - News articles about management changes

- MUC-7 (1997)
  - News articles about space vehicle and missile launches

MUC4 Text

SAN SALVADOR, 26 APR 89 (EL DIARIO DE HOY) -- [TEXT]

PRESIDENT-ELECT ALFREDO CRISTIANI YESTERDAY ANNOUNCED CHANGES IN THE ARMY'S STRATEGY TOWARD URBAN TERRORISM AND THE FARABUNDO MARTI NATIONAL LIBERATION FRONT'S (FMLN) DIPLOMATIC OFFENSIVE TO ISOLATE THE NEW GOVERNMENT ABROAD.

CRISTIANI SAID: "WE MUST ADJUST OUR POLITICAL-MILITARY STRATEGY AND MODIFY LAWS TO ALLOW US TO PROFESSIONALLY COUNTER THE FMLN'S STRATEGY."

AS THE PRESIDENT-ELECT WAS MAKING THIS STATEMENT, HE LEARNED ABOUT THE ASSASSINATION OF ATTORNEY GENERAL ROBERTO GARCIA ALVARADO. AS PUBLISHED, ALVARADO WAS KILLED BY A BOMB PRESUMABLY PLACED BY AN URBAN GUERRILLA GROUP ON TOP OF HIS ARMORED VEHICLE AS IT STOPPED AT AN INTERSECTION IN SAN MIGUELITO NEIGHBORHOOD, NORTH OF THE CAPITAL.

Template Details

- The template consists of 25 fields.

- Four different types:
  1. String slots (e.g. 6): filled using strings extracted from text
  2. Text conversion slots (e.g. 4): inferred from the document
  3. Set Fill Slots (e.g. 14): filled with a finite, fixed set of possible values
  4. Event identifiers (0 and 1): store some identifier information
Information Extraction and Weakly-supervised Learning

**MUC6 Example**

McCann has initiated a new so-called global collaborative system, composed of world-wide account directors paired with creative partners. In addition, Peter Kim was hired from WPP Group’s J. Walter Thompson last September as vice chairman, chief strategy officer, world-wide.

**Evaluation metrics**

- **Aim of evaluation is to work out whether the system can identify the events in the gold standard and no extra ones**

- **Gold standard**
  - True Positives
  - False Positives

- **System**
  - True Positives
  - False Positives

- **False negatives**

**Precision**

- **A system’s precision score measures the number of events identified which are correct**

  Precision (P) = Correct Answers / Answers Produced = True Positives / (True Positives + False Positives)

  - Ranges between 0 (all of the identified events were incorrect) and 1 (all of them were correct)

**Recall**

- **Recall score measures the number of correct events which were identified**

  Recall (R) = Correct Answers / Total Possible Correct = True Positives / (True Positives + False Negatives)

  - Ranges between 0 (no correct events identified) and 1 (all of the correct events were identified)

**Examples**

- **High Precision, low Recall**

- **Gold standard**
  - System

- **High Recall, low Precision**

- **Gold standard**
  - System
**F-measure**

- Precision and recall are often combined into a single metric: F-measure

\[ F = \frac{2PR}{P + R} \]

**System Performance**

- Performance of best systems in various MUCs

<table>
<thead>
<tr>
<th>Evaluation/Named Entity</th>
<th>Scenario Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUC3</td>
<td>R &lt; 0.5, P &lt; 0.7</td>
</tr>
<tr>
<td>MUC4</td>
<td>F &lt; 0.53</td>
</tr>
<tr>
<td>MUC5</td>
<td>F &lt; 0.53</td>
</tr>
<tr>
<td>MUC6</td>
<td>F &lt; 0.97</td>
</tr>
<tr>
<td>MUC7</td>
<td>F &lt; 0.94</td>
</tr>
</tbody>
</table>

**Summary**

- Information Extraction is the process of identifying specific pieces of information from text
- Normally carried out as a two-stage process:
  1. Name identification
  2. Event extraction
- Message Understanding Conferences are the best-known IE evaluation
- Most commonly used evaluation metrics are precision, recall and F-measure
- This course concentrates on machine learning approaches to event extraction

**Part 2: Relation Identification**

**AutoSlog: Overview**

- Constructing “concept dictionary” for IE task
  - Here concept dictionary means extraction patterns
  - Lexicon (words and terms) is another knowledge base
- Uses a manually tagged corpus
  - MUC-4: Terrorist attacks in Latin America
  - Names of perpetrator, victim, instrument, site, ...
- Method: “Selective concept extraction”
  - Shallow sentence analyzer (partial parsing)
  - Selective semantic analyzer
  - Uses a “dictionary of concept nodes”

**Riloff 1993**

Automatically Constructing a Dictionary for Information Extraction Tasks
Information Extraction and Weakly-supervised Learning

**Concept node**

Has the following elements:

- **A triggering lexical item**
  - E.g., "diplomat was kidnapped"
  - "kidnapped" can trigger an active or the passive node
- **Enabling conditions (in the context)**
  - E.g., passive context: match on "was/were kidnapped"
- **Case frame**
  - The set of slots to fill/extract from surrounding context
  - Each slot has *selectional restrictions* for the filler
  - (hard/soft constraints?)

**Application**

- **Input sentence:**
  - "the mayor was kidnapped"

**Template:**

- TerrorAttack:
  - Perpetrator: __________
  - Victim: ___________
  - Instrument: __________
  - Site: ___________
  - Date: ___________

- **MUC-4 (1992) UMASS system contained**
  - 5426 lexical entries, with semantic class information
  - 389 concept node definitions/templates
  - 1500 person/hours to build

**MUC-4 task**

- Extract zero or more events for each document
  - event = filled template = large case frame
- **Slots:**
  - perpetrator, instrument
  - human target, physical target,
  - site, date
- **Training corpus**
  - 1500 documents (a lot!)
  - + answer keys
  - Extracted by keyword search (IR) from newswire
  - 50% relevant

**Heuristics**

- **Slot fill**
  - *First reference to the slot fill is likely to specify the relationship of the slot fill to the event*
  - Surrounding context of the first reference contains words or phrases that specify the relationship of the slot fill to the event
  - *(A little strong?)*

**AutoSlog: Algorithm**

- **Given filled templates**
- **For each slot fill:**
  - Find *first* reference to a fill
  - Shallow parsing/semantic analysis of sentence (CIRCUS shallow analyzer)
  - Find *conceptual anchor point:*
    - Trigger word = word that will activate the concept
    - Find conditions
    - Build concept node definition
- **Usually assume the verb will determine the role of the NP**

**Syntactic heuristics**

- **patterns:** <matched fill>  examples: <slot> context trigger
  
  - <subject> passive-verb
  - <subject> active-verb
  - <subject> verb infinitive
  - <subject> auxiliary noun
  
  - passive-verb <dobj>:
    - killed <victim> was murdered
    - battered <target>
  - active-verb <dobj>:
    - to *kill <victim>*
  - verb infinitive <dobj>:
    - threatened to *attack <target>*
    - killing <victim>
  - gerund <dobj>:
    - fatality was <victim>
  - noun auxiliary <dobj>:
    - bomb against <target>
  - noun prep <np>:
    - *killed with <instrument>*
  - passive-verb prep <np>:
    - was *aimed at <target>*
Information Extraction and Weakly-supervised Learning

Concept node definition

Id: DEV-MUC4-0657
Slot filler: “public buildings”
Sentence: (in la oroya, junin department, in the central peruvian mountain range, public buildings were bombed and a car-bomb was detonated.)

CONCEPT NODE
Name: target-subject-passive-verb-bombed
Trigger: bombed
Variable Slots: (target (*$^ 1))
Constraints: (class phys-target *$*)
Constant Slots: (type bombing)
Enabling Conditions: (passive)

Concept node: not so good

Id: DEV-MUC4-1192
Slot filler: “gilberto molasco”
Sentence: (they took 2-year-old gilberto molasco, son of patricio rodriguez, and 17-year-old andres argueta, son of emenesto argueta.)

CONCEPT NODE
Name: victim-active-verb-dobj-took
Trigger: took
Variable Slots: (victim (*DORF* 1))
Constraints: (class victim *DORF*)
Constant Slots: (type kidnapping)
Enabling Conditions: (active)

Problems

• When “first-mention” heuristic fails
• When syntactic heuristic finds wrong trigger
• When shallow parser fails

Introduce human in the loop to filter out bad concept nodes

Results

• 1500 texts, 1258 answer keys (templates)
• 4780 slot fillers (only 6 slot types)
• AutoSlog generated 1237 concept nodes
• After human filtering: 450 concept nodes
  Final concept node dictionary
• Compare to manually-built dictionary
• Run real MUC-4 IE task

Results

• Two tests: TST3 and TST4
• Official MUC-4/TST4 includes (!) 76 concepts found by AutoSlog
  Difference could be even greater
• Comparable to manually-trained system

<table>
<thead>
<tr>
<th>System/Test Set</th>
<th>Recall</th>
<th>Precision</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUC-4/TST3</td>
<td>46</td>
<td>56</td>
<td>50.51</td>
</tr>
<tr>
<td>AutoSlog/TST3</td>
<td>43</td>
<td>56</td>
<td>48.65</td>
</tr>
<tr>
<td>MUC-4/TST4</td>
<td>44</td>
<td>40</td>
<td>41.90</td>
</tr>
<tr>
<td>AutoSlog/TST4</td>
<td>39</td>
<td>45</td>
<td>41.79</td>
</tr>
</tbody>
</table>
Automatically Generating Extraction Patterns from Untagged Text

**Introduction**
- Construct “dictionary” of patterns for IE
- AutoSlog performance comparable to human
  - Required annotated corpus—an expensive proposition
- Other competition:
  - PALKA (Kim & Moldovan, 1993)
  - CRYSTAL (Soderland, 1995)
  - LIEP (Huffman, 1996)
- Can we do without annotated corpus?
  - AutoSlog-TS
    - Generates extraction patterns
    - No annotated corpus
    - Needs only classified corpus: relevant vs. non-relevant

**Example**
- Input sentence: Ricardo Castellar, the mayor, was kidnapped yesterday by the FMLN.
- Partial parse:
  - Ricardo Castellar = subject
- Pattern:
  - <victim> was kidnapped
- Select the verb as trigger (usually)
- May produce bad patterns
  - Person in the loop corrects bad patterns – fast
- Problem: annotation is slow

**Annotation is hard**
- Annotating toy examples is easy
- Real data: what should be annotated?
- Instances (NPs) have many problems:
  - Include modifiers or only head noun?
  - Meaning of head noun may depend heavily on modifiers
  - All modifiers or only some?
  - Determiners?
  - If part of a conjunction: all conjuncts or only one?
  - Appositive? Prepositional phrases?
- Difficult to set guidelines that cover every instance
- Without guidelines, data will be inconsistent

**AutoSlog-TS**
- Life would be easier if we did not have to worry about annotation
- When AutoSlog had annotations for slots, it generated annotations for the NPs it found in the slots
- New Idea: exhaustive processing:
  - Generate an extraction pattern for each noun phrase in training corpus
  - Tens of thousands of patterns
  - Much more than with AutoSlog
  - Evaluate patterns based on co-occurrence statistics with relevant sub-corpus
  - Choose patterns that are correlated with the relevant sub-corpus

**Process**

Stage 1
- Sentence Analyzer
- World Trade Center
- AutoSlog Heuristics
- Concept Nodes: <xx> was bombed by <yy>

Stage 2
- Sentence Analyzer
- Concept Node Dictionary: <xx> was bombed by <yy>
- RELS: <xx> was killed by <yy>
**Syntactic heuristics**

- Pattern: `<match fill>`  
  Example: `<slot>` context trigger

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;subject&gt;</code> passive-verb <code>&lt;victim&gt;</code> was murdered</td>
<td><code>&lt;perpetrator&gt;</code> bombed <code>&lt;target&gt;</code> attempted to kill <code>&lt;victim&gt;</code> was <code>victim</code></td>
</tr>
<tr>
<td><code>&lt;subject&gt;</code> active-verb <code>&lt;perpetrator&gt;</code> bombed <code>&lt;target&gt;</code></td>
<td><code>&lt;noun auxiliary&gt;</code> killed <code>&lt;victim&gt;</code></td>
</tr>
<tr>
<td><code>&lt;subject&gt;</code> verb infinitive <code>&lt;target&gt;</code> threatened to attack <code>&lt;victim&gt;</code></td>
<td><code>&lt;noun prep&gt;</code> killed with <code>&lt;instrument&gt;</code></td>
</tr>
<tr>
<td><code>&lt;subject&gt;</code> auxiliary noun <code>fatalty</code> was <code>&lt;victim&gt;</code></td>
<td><code>&lt;active-verb prep&gt;</code> bomb against <code>&lt;victim&gt;</code></td>
</tr>
<tr>
<td>passive-verb <code>&lt;dobj&gt;</code> killed <code>&lt;victim&gt;</code></td>
<td><code>&lt;active-verb prep&gt;</code> was <code>aimed at</code> <code>&lt;target&gt;</code></td>
</tr>
<tr>
<td>active-verb <code>&lt;dobj&gt;</code> to kill <code>&lt;victim&gt;</code></td>
<td>infinitive <code>&lt;dobj&gt;</code></td>
</tr>
<tr>
<td>gerund <code>&lt;dobj&gt;</code> killing <code>&lt;victim&gt;</code></td>
<td>noun auxiliary <code>&lt;dobj&gt;</code></td>
</tr>
</tbody>
</table>

**What is new**

- Two new pattern heuristics:
  - `<subject>` active-verb dobj (*)
  - infinitive prep `<np>`
- More than one pattern may fire (*)
  - Relevance determines whether prefer longer or shorter pattern (matching subject or dobj, respectively)
- Pattern relevance is modeled by conditional probability:
  \[ Pr(\text{relevant document} \mid \text{pattern matched}) = \frac{\text{relevant-frequency}}{\text{overall-frequency}} \]

**Main idea**

- Domain-specific expressions will appear more often in the relevant documents than in non-relevant ones
- Don’t want to use just unconditional probability
- Rank patterns in order of relevance
- Patterns with relevance(p) < 0.5 are discarded
- **Score(p) = Relevance(p) * log support(p)**
  - Support = how many times p occurs in training corpus
  - Somewhat ad hoc measure, but works ok

**Experiments**

- Manually inspect performance on MUC-4
- AutoSlog:
  - Used the 772 relevant documents of 1500 training set
  - Produced 1237 patterns, manually inspect in 5 hours
  - Final dictionary: 450 patterns
- AutoSlog-TS:
  - Generated 32,345 distinct patterns
  - Discard patterns that appear once: 11,225 patterns
  - Rank according to score: top 25 patterns

**Top-ranked 25 patterns**

1. `<subj>` exploded
2. `murder of <np>`
3. `assassination of <np>`
4. `<subj>` was located
5. `<subj>` was kidnapped
6. `attack on <np>`
7. `<subj>` was injured
8. `exploded in <np>`
9. `death of <np>`
10. `<subj>` killed
11. `<subj>` kidnapped
12. caused `<dobj>`
13. `<subj>` was wounded

**User review**

- User judged pattern relevance
- Assign category to accepted patterns
  - This was automatic in AutoSlog, because of annotation
- Of 1970 top-ranked patterns, kept 210
  - After 1970 quit: few patterns were being accepted
  - Reviewed in 85 min: quicker than AutoSlog
  - Much smaller dictionary than AutoSlog (450)
- Kept only patterns for
  - Perpetrator, victim, target, weapon
  - Not for location (excluded “exploded in `<np>`”)
- Evaluate
**Evaluation**

- NP extracted by accepted pattern can be:
  - Correct
  - Duplicate: coreferent in text with an item in key
  - Mislabeled: incorrect
  - Missing: in key but not in response
  - Spurious: in response but not in key

- Compare with AutoSlog:
  - t-test
  - Significant improvement for AutoSlog-TS in spurious
  - No significant difference in others

**IE measures:**

- Recall = cor / (cor+mis)
- Precision = (cor+dup) / (cor+dup+inc+spu)
- AutoSlog-TS slightly lower recall, but better precision \( \rightarrow \) higher F

**Final analysis**

- AutoSlog passed through more low-relevance patterns, got higher recall, but poor precision
  - AutoSlog-TS filtered low-ranked patterns, with low relevance
  - AutoSlog-TS produced 158 patterns with Rel(p) > .90
  - Only 45 of these were among AutoSlog 450 patterns
  - E.g.: AutoSlog accepted pattern "<subject> admitted"
  - AutoSlog-TS assigned it negative correlation: 46%
  - But if used pattern "<subject> admitted responsibility"...

**Conclusion**

- AutoSlog-TS reduces user involvement in porting IE system to new domain. The human:
  - Provides texts classified as relevant–irrelevant
  - Judges resulting ranked list of patterns
  - Labels resulting patterns (what kind of event template they will generate)

---

**Table 1: AutoSlog Results**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Perp</td>
<td>30</td>
<td>22</td>
<td>1</td>
<td>11</td>
<td>129</td>
</tr>
<tr>
<td>Victim</td>
<td>41</td>
<td>24</td>
<td>7</td>
<td>18</td>
<td>113</td>
</tr>
<tr>
<td>Target</td>
<td>39</td>
<td>19</td>
<td>8</td>
<td>18</td>
<td>108</td>
</tr>
<tr>
<td>Total</td>
<td>116</td>
<td>65</td>
<td>16</td>
<td>47</td>
<td>350</td>
</tr>
</tbody>
</table>

**Table 2: AutoSlog-TS Results**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Perp</td>
<td>30</td>
<td>27</td>
<td>2</td>
<td>12</td>
<td>97</td>
</tr>
<tr>
<td>Victim</td>
<td>40</td>
<td>25</td>
<td>7</td>
<td>19</td>
<td>85</td>
</tr>
<tr>
<td>Target</td>
<td>32</td>
<td>23</td>
<td>17</td>
<td>16</td>
<td>58</td>
</tr>
<tr>
<td>Total</td>
<td>102</td>
<td>75</td>
<td>26</td>
<td>47</td>
<td>240</td>
</tr>
</tbody>
</table>

---

Yangarber, Grishman, Tapanainen, Huttunen 2000

Acquisition of semantic patterns for IE
**Trend in knowledge acquisition**

- build patterns from examples: manual
  - Yangarber '97
- generalize from multiple examples: annotated corpus
  - Crystal, Whisk (Soderland), Rapier (Califf)
- active learning: reduce amount of annotation
  - Soderland '99, Califf '99
- automatic learning: corpus with relevance judgements
  - Riloff '96
- co-learning/booststrapping
  - Brin '98, Agichtein '00

**Learning event patterns: Goals**

- Minimize manual labor required to construct pattern base for new domain
  - un-annotated text
  - un-classified text
  - un-supervised learning
- Use very large corpora — larger than we could ever tag manually — to boost coverage

---

**Principle I: Density**

- Density of Pattern Distribution:
  - If we have relevance judgements for documents in a corpus, for the given task,
  - then the patterns which are much more frequent in relevant documents than overall will generally be good patterns
  - Riloff (1996) finds patterns related to terrorist attacks

**Density Criterion**

- $U$ — universe of all documents
- $R$ — set of relevant documents
- $H(p)$ — set of documents where pattern $p$ matched

$$\frac{|H \cap R|}{|H|} \gg \frac{|R|}{|U|}$$

**Principle II: Duality**

- Duality between patterns and documents:
  - relevant documents are strong indicators of good patterns
  - good patterns are strong indicators of relevant documents

**ExDisco: Outline**

- Initial query: a small set of seed patterns which partially characterize the topic of interest
- Retrieve documents containing seed patterns: "relevant documents"
- Rank patterns in relevant documents by frequency in relevant docs vs. overall frequency
- Add top-ranked pattern to seed pattern set
Note

- Go back to look for relevant documents, but with the new, enlarged patterns
- In this way, pattern set and document set grow in tandem

But: What is a pattern?

Methodology

- Problems:
  - Pre-processing
  - Pattern ranking and document relevance

Pre-processing: NEs

- Begin with several pre-processing steps
- For each document, find and classify all proper names:
  - Person
  - Location
  - Organization
  - …
- Replace each name with its category label
  - <Per> <Org> <Loc> …
- Factor out unnecessary distinctions in text
  - To maximize redundancy

Pre-processing: syntax

- Parse document
  - Full parse
  - Regularize: passive clauses, relative clauses, etc. \(\Rightarrow\) common form (active clause)
  - “John, who was hired by IBM” \(\Rightarrow\) “IBM hire John”
- For each clause, collect a candidate pattern: tuple: heads of
  - Subject
  - Verb
  - Direct object
  - Object/subject complement
  - Locative and temporal modifiers
  - …

Proper Names are hard too

- Person
- Location
- Organization
  - “IBM”, “Sony, Ltd.”, “Calvin Klein &Co”, “Calvin Klein”
- Products/Artifacts/Works of Art
  - “DC-10”, “SCUD”, “Barbie”, “Barney”, “Gone with the Wind”, “Mona Lisa”
- Other groups
- Laws, Regulations, Legal Cases
- Major Events, political, meteorological, etc.
  - “Hurricane George”, “El Niño”, “Million Man March”, “Great Depression”

Pre-processing: syntax

- Clause \(\Rightarrow\) [subject, verb, object]
  - Primary tuple
- May still not appear with sufficient frequency
**Pre-processing**

- Tuples → generalized patterns
  - [Subject Verb Object]

**Scoring Patterns**

- \( \text{Score}(p) = \frac{|R \cap H|}{|H|} \cdot \text{Log relevant document count} \)
- "Binary" support: \( \text{Sup}(p) = |R \cap H| \)
- \( \text{Score}(p) = \frac{\text{Sup}(p)}{|H|} \cdot \log \text{Sup}(p) \)
- Accept highest-scoring pattern
Strength of relevance

- If patterns and documents are accepted unconditionally, algorithm will quickly start learning non-relevant documents and patterns
  - E.g., “person died”
- Need to introduce probabilistic model of pattern goodness and document relevance

Weighted pattern “goodness”

- When seed pattern matches a document, the document is considered 100% relevant
- Discovered patterns are considered less certain, relevance (weight of the match) is between 0 and 1

\[
\text{Prec}^{d+1}(p) = \frac{1}{|H(p)|} \cdot \sum_{d \in H(p)} \text{Rel}^d(d)
\]

- Documents containing them are considered partially relevant
- “Internal” graded document relevance
  - (rather than binary)

Graded Document Relevance

- Disjunctive voting

\[
\text{Rel}^{d+1}(d) = 1 - \prod_{p \in \mathcal{H}^d} \left(1 - \text{Prec}^{d+1}(p)\right)
\]

- Continuous

\[
\text{Rel}(d) = 1 - \sqrt[|\mathcal{H}(d)|]{\prod_{p \in \mathcal{H}(d)} \left(1 - \text{Prec}(p)\right)^{w_p}}
\]

\[
\text{Sup}(p) = \sum_{d \in \mathcal{H}(p)} \text{Rel}(d)
\]

\[
w_p = \log \text{Sup}(p)
\]

Duality

- Mutual recursion

\[
\text{Prec}^{d+1}(p) = \frac{1}{|H(p)|} \cdot \sum_{d \in H(p)} \text{Rel}^d(d)
\]

- and

\[
\text{Rel}^{d+1}(d) = 1 - \prod_{p \in \mathcal{H}^d} \left(1 - \text{Prec}^{d+1}(p)\right)
\]

\[
\text{Rel}(d) = 1 - \sqrt[|\mathcal{H}(d)|]{\prod_{p \in \mathcal{H}(d)} \left(1 - \text{Prec}(p)\right)^{w_p}}
\]

Evaluation

- Qualitative:
  - Look at discovered patterns
    (New patterns, missed in manual building)
- Quantitative:
  - Document filtering
  - Slot filling

Experiments

- Scenario: Management succession
  - as in MUC-6
- Scenario: Corporate Mergers & Acquisitions
### Management Succession

- Source: Wall Street Journal
- Training corpus: ~10,000 articles (9,224)

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>company</td>
<td>v-appoint</td>
<td>person</td>
</tr>
<tr>
<td>person</td>
<td>v-resign</td>
<td>-</td>
</tr>
<tr>
<td>person</td>
<td>succeed</td>
<td>person</td>
</tr>
<tr>
<td>person</td>
<td>be, become</td>
<td>president, officer, executive</td>
</tr>
<tr>
<td>person</td>
<td>retire</td>
<td>-</td>
</tr>
<tr>
<td>company</td>
<td>name</td>
<td>president, successor</td>
</tr>
<tr>
<td>person</td>
<td>join, head run, start leave, own</td>
<td>company</td>
</tr>
<tr>
<td>person</td>
<td>serve</td>
<td>board, company sentence</td>
</tr>
<tr>
<td>person</td>
<td>hold, resign fill, retain</td>
<td>position</td>
</tr>
<tr>
<td>person</td>
<td>relinquish leave, assume hold, accept retain, take</td>
<td>post</td>
</tr>
</tbody>
</table>

### Seed: Management

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>company</td>
<td>v-appoint</td>
<td>person</td>
</tr>
<tr>
<td>person</td>
<td>v-resign</td>
<td>-</td>
</tr>
</tbody>
</table>

- v-appoint = (appoint, elect, promote, name, nominate)
- v-resign = (resign, depart, quit)

- Run ExDisco for ~80 iterations

### Note

- ExDisco also finds classes of terms, in tandem with patterns and documents
- These will be useful
- Discovers new patterns, not found in manual search for patterns

### Evaluation: new patterns

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
<th>Complements</th>
</tr>
</thead>
<tbody>
<tr>
<td>company</td>
<td>bring</td>
<td>person</td>
<td>[as+officer]</td>
</tr>
<tr>
<td>person</td>
<td>come</td>
<td>-</td>
<td>[to+company]</td>
</tr>
<tr>
<td>person</td>
<td>return</td>
<td>-</td>
<td>[as+officer]</td>
</tr>
<tr>
<td>person</td>
<td>rejoin</td>
<td>company</td>
<td>[as+officer]</td>
</tr>
<tr>
<td>person</td>
<td>continue</td>
<td>remain</td>
<td>[as+officer]</td>
</tr>
<tr>
<td>person</td>
<td>replace</td>
<td>person</td>
<td>[as+officer]</td>
</tr>
<tr>
<td>person</td>
<td>pursue</td>
<td>interest</td>
<td>-</td>
</tr>
</tbody>
</table>

- New patterns: not found in manual training

### Mergers & Acquisitions

- Source: Associated Press (AP)
- Training corpus: ~14,000 articles
- ~3 months from 1989
**Seed: Acquisitions**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>* v-buy</td>
<td>c-company</td>
<td>merge *</td>
</tr>
</tbody>
</table>

- v-buy = {buy, purchase}

**Natural Disasters**

- Source: Associated Press
- Training corpus: ~14,000 articles
- Test corpus:
  - n/a

**Natural Disaster: seed**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>n-disaster</td>
<td>cause *</td>
<td></td>
</tr>
<tr>
<td>n-disaster</td>
<td>v-damage n-structure</td>
<td></td>
</tr>
</tbody>
</table>

- n-disaster = {earthquake, tornado, flood, hurricane, landslide, snowstorm, avalanche}
- v-damage = {damage, hit, destroy, ravage}
- v-structure = {street, bridge, house, home, -}

- Run discovery procedure

**Discovered patterns**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>n-disaster</td>
<td>cause</td>
<td>*</td>
</tr>
<tr>
<td>n-disaster</td>
<td>v-damage n-structure</td>
<td></td>
</tr>
<tr>
<td>quake</td>
<td>register &lt;number&gt; measure</td>
<td></td>
</tr>
<tr>
<td>quake</td>
<td>was felt</td>
<td></td>
</tr>
<tr>
<td>storm</td>
<td>quake knock-out power</td>
<td></td>
</tr>
<tr>
<td>aftershock</td>
<td>quake injure kill people</td>
<td></td>
</tr>
<tr>
<td>it</td>
<td>cause damage</td>
<td></td>
</tr>
<tr>
<td>quake</td>
<td>strike</td>
<td></td>
</tr>
</tbody>
</table>

**Task: Corporate Lawsuits**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>* v-sue</td>
<td>organization</td>
<td>bring suit</td>
</tr>
</tbody>
</table>

- v-sue = { sue, litigate }

- Run discovery procedure
**Discovered patterns**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Verb</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>v-sue</td>
<td>organization</td>
</tr>
<tr>
<td>*</td>
<td>bring</td>
<td>suit</td>
</tr>
<tr>
<td>organization</td>
<td>file</td>
<td>suit</td>
</tr>
<tr>
<td>plaintiff</td>
<td>seek</td>
<td>damages</td>
</tr>
<tr>
<td>person</td>
<td>hear</td>
<td>case</td>
</tr>
<tr>
<td>company</td>
<td>deny</td>
<td>allegation</td>
</tr>
<tr>
<td>person</td>
<td>reject</td>
<td>argument</td>
</tr>
<tr>
<td>company</td>
<td>appeal</td>
<td>-</td>
</tr>
<tr>
<td>company</td>
<td>settle</td>
<td>charge</td>
</tr>
</tbody>
</table>

**Evaluation: Text Filtering**

- How effective are discovered patterns at selecting relevant documents?
  - Indirect evaluation
  - Similar to MUC text filtering task
  - IR-style evaluation
  - Documents matching at least one pattern

**Performance:**

<table>
<thead>
<tr>
<th>Pattern set</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seed</td>
<td>15%</td>
<td>88%</td>
</tr>
<tr>
<td>Seed+discovered</td>
<td>79%</td>
<td>78% (85)</td>
</tr>
</tbody>
</table>

**Text filtering**

- On each iteration each document has internal measure of relevance
- Determine external relevance:

\[ \text{Rel}^*_p(d) = \begin{cases} 1 & \text{if } \text{Rel}_i(d) \geq \theta \\ 0 & \text{if } \text{Rel}_i(d) < \theta \end{cases} \]

- \( \theta = 0.5 \)
- Each document is rated “relevant” or “non-relevant”
- Compare to correct answer
- Measure recall and precision

**Management Succession**

- Source: Wall Street Journal
- Training corpus: ~10,000 articles (9,224)
- Test corpora:
  - 100 docs: MUC–6 Development corpus
  - 100 docs: MUC–6 Formal Evaluation corpus
  - relevance judgments and filled templates
**Evaluation: Slot filling**

- How effective are patterns within a complete IE system?
- MUC–style IE on MUC–6 corpora

<table>
<thead>
<tr>
<th>pattern base</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seed</td>
<td>38</td>
<td>83</td>
<td>52.60</td>
</tr>
<tr>
<td>ExDisco</td>
<td>62</td>
<td>80</td>
<td>69.94</td>
</tr>
<tr>
<td>Union</td>
<td>69</td>
<td>79</td>
<td>73.50</td>
</tr>
<tr>
<td>Manual–MUC</td>
<td>54</td>
<td>71</td>
<td>61.93</td>
</tr>
<tr>
<td>Manual–Now</td>
<td>69</td>
<td>79</td>
<td>73.91</td>
</tr>
</tbody>
</table>

---

**Evaluation: Slot filling**

- How effective are patterns within a complete IE system?
- MUC–style IE on MUC–6 corpora

<table>
<thead>
<tr>
<th>pattern base</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seed</td>
<td>38</td>
<td>83</td>
<td>52.60</td>
<td>27</td>
<td>74</td>
<td>39.58</td>
</tr>
<tr>
<td>ExDisco</td>
<td>62</td>
<td>80</td>
<td>69.94</td>
<td>52</td>
<td>72</td>
<td>60.16</td>
</tr>
<tr>
<td>Union</td>
<td>69</td>
<td>79</td>
<td>73.50</td>
<td>57</td>
<td>73</td>
<td>63.56</td>
</tr>
<tr>
<td>Manual–MUC</td>
<td>54</td>
<td>71</td>
<td>61.93</td>
<td>47</td>
<td>70</td>
<td>56.40</td>
</tr>
<tr>
<td>Manual–Now</td>
<td>69</td>
<td>79</td>
<td>73.91</td>
<td>56</td>
<td>75</td>
<td>64.04</td>
</tr>
</tbody>
</table>

---

**Mergers & Acquisitions**

- Source: Associated Press (AP)
- Training corpus: ~ 14,000 articles
  - ~ 3 months from 1989
- Test corpus:
  - 200 documents,
  - retrieved by keywords
  - relevance judged manually

---

**Acquisitions: text filtering**

**Evaluation: Slot filling**

- How effective are patterns within a complete IE system?
- MUC–style IE on MUC–6 corpora

<table>
<thead>
<tr>
<th>training</th>
<th>pattern base</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Seed</td>
<td>38</td>
<td>83</td>
<td>52.60</td>
<td>27</td>
<td>74</td>
<td>39.58</td>
</tr>
<tr>
<td></td>
<td>ExDisco</td>
<td>62</td>
<td>80</td>
<td>69.94</td>
<td>52</td>
<td>72</td>
<td>60.16</td>
</tr>
<tr>
<td></td>
<td>Union</td>
<td>69</td>
<td>79</td>
<td>73.50</td>
<td>57</td>
<td>73</td>
<td>63.56</td>
</tr>
<tr>
<td></td>
<td>Manual–MUC</td>
<td>54</td>
<td>71</td>
<td>61.93</td>
<td>47</td>
<td>70</td>
<td>56.40</td>
</tr>
<tr>
<td></td>
<td>Manual–Now</td>
<td>69</td>
<td>79</td>
<td>73.91</td>
<td>56</td>
<td>75</td>
<td>64.04</td>
</tr>
</tbody>
</table>

---

**Information Extraction and Weakly-supervised Learning**

**Mergers & Acquisitions**

- Source: Associated Press (AP)
- Training corpus: ~ 14,000 articles
  - ~ 3 months from 1989
- Test corpus:
  - 200 documents,
  - retrieved by keywords
  - relevance judged manually

---

**Acquisitions: text filtering**

**Evaluation: Slot filling**

- How effective are patterns within a complete IE system?
- MUC–style IE on MUC–6 corpora

<table>
<thead>
<tr>
<th>training</th>
<th>pattern base</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
<th>recall</th>
<th>precision</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Seed</td>
<td>38</td>
<td>83</td>
<td>52.60</td>
<td>27</td>
<td>74</td>
<td>39.58</td>
</tr>
<tr>
<td></td>
<td>ExDisco</td>
<td>62</td>
<td>80</td>
<td>69.94</td>
<td>52</td>
<td>72</td>
<td>60.16</td>
</tr>
<tr>
<td></td>
<td>Union</td>
<td>69</td>
<td>79</td>
<td>73.50</td>
<td>57</td>
<td>73</td>
<td>63.56</td>
</tr>
<tr>
<td></td>
<td>Manual–MUC</td>
<td>54</td>
<td>71</td>
<td>61.93</td>
<td>47</td>
<td>70</td>
<td>56.40</td>
</tr>
<tr>
<td></td>
<td>Manual–Now</td>
<td>69</td>
<td>79</td>
<td>73.91</td>
<td>56</td>
<td>75</td>
<td>64.04</td>
</tr>
</tbody>
</table>
**Automatic discovery**

- ExDisco performance within range of human performance on text filtering (4-week development)
- From un-annotated text: allows us to take advantage of very large corpora
  - Redundancy
  - Duality
- Limited user intervention

**Summary**

- Discover patterns
- Indirect evaluation
  - Via text filtering
- Maintains internal model of pattern precision and document relevance
  - Rather than binary judgments

**Preview**

- Investigate different extraction scenarios
  - Variation in Recall/Precision curves
  - Due to seed quality
  - Due to inherent properties of scenario
- Utilize "peripheral" clausal arguments
- Discover Noun-Phrase patterns
- Discovery for other knowledge bases
  - word classes
  - template mappings

**Yangarber 2003**

**Counter-training**

**Prior Work**

On knowledge acquisition:
- Yangarber, Grishman, Tapanainen, Huttunen (2000), others
  - Algorithm does not know when to stop iterating
  - Needs review by human, supervised or ad hoc thresholds
- Yangarber, Lin, Grishman (2002)
  - Natural convergence

**Counter-training**

- Train several learners simultaneously
- Compete with each other in different domains
- Improve precision
- Provide indication to each other when to stop learning
Algorithm: Pre-processing

- Factor out NEs (and other OOVs)
  - RE grammar
- Parse
  - General-purpose dependency parser
- Tree normalization
  - Passive → active
- Pattern extraction
  - Tree → core constituents [Company hire Person]

Bootstrap Learner: ExDisco

- Initial query:
  - A small set of seed patterns which partially characterize topic of interest
- Retrieve documents containing seed patterns:
  - “Relevant documents”
- Rank patterns (in relevant documents)
  - According to frequency in relevant docs vs. Overall frequency
- Add top-ranked pattern to seed pattern set

Pattern score

- Trade-off recall and precision
- Eventually mono-learner will pick up non-specific patterns
  - Match documents relevant to the scenario, but also match non-relevant documents

Counter-training

- Introduce multiple learners in parallel
- Learning in different, competing categories
- Documents which are “ambiguous” will receive high relevance score in more than one scenario
- Prevent learning patterns which match such “ambiguous” documents

Refine precision

- Pattern precision measure takes into account negative evidence provided by other learners.
- Continue as long as number of scenarios/categories that are still acquiring patterns is > 1
  - When = 1, we are back to the mono-training case...
Information Extraction and Weakly-supervised Learning

Experiments

- Corpus:
  - 15,000 documents
- Test:
  - MUC–6 training data (management succession)
  - + 150 documents tagged manually (M&A)

Scenarios/categories to compete

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Seed Patterns</th>
<th>Documents</th>
<th>Last</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management Succession</td>
<td>[Company appoint Person] [Person quit]</td>
<td>220</td>
<td>143</td>
</tr>
<tr>
<td>Mergers &amp; Acquisitions</td>
<td>[Buy Company] [Company merge]</td>
<td>231</td>
<td>230</td>
</tr>
<tr>
<td>Legal Action</td>
<td>[lawyer Organization] [bring settle suit]</td>
<td>109</td>
<td>132</td>
</tr>
<tr>
<td>Bill/Law Passing</td>
<td>[pass bill]</td>
<td>89</td>
<td>79</td>
</tr>
<tr>
<td>Political Election</td>
<td>[make/write/open election/campaign]</td>
<td>42</td>
<td>24</td>
</tr>
<tr>
<td>Sports Event</td>
<td>[ran/win/lose competition/event]</td>
<td>25</td>
<td>19</td>
</tr>
<tr>
<td>Layoff</td>
<td>[appoint/release layoff]</td>
<td>43</td>
<td>15</td>
</tr>
<tr>
<td>Bankruptcy</td>
<td>[file/declare bankruptcy]</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Natural Disaster</td>
<td>[declare bankruptcy]</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>Other</td>
<td>[corporate/cut/layoff people/property]</td>
<td>413</td>
<td>-</td>
</tr>
</tbody>
</table>

Counter-training

- Train several learners simultaneously
- Compete with each other in different domains
- Improve precision
- Provide indication to each other when to stop learning

Current Work

- Choice of seeds
- Choice of scenarios
  - Corpus representation
- Ambiguity
  - At document level
  - At pattern level
- Apply to IE customization tasks
Information Extraction and Weakly-supervised Learning

General framework

Bootstrapping approaches

General procedure

• Builds up a learner/classifier
  – Set of rules
  – To identify a set of datapoints as members of a category
• Objective: find set of rules that partitions the dataset into "relevant" vs "non-relevant" w.r.t. the category
• Rules = contextual patterns

Features of the problem

• Duality between instance space and rule space
• Many–many
  – More than one rule applies to a datapoint
  – More than one datapoint is identified by a rule
• Redundancy
  – Good rules indicate relevant datapoints
  – Relevant datapoints indicate good rules
• If these criteria are met, method may apply

Counter-training framework

• Pre-process large corpus
  – Factor out irrelevant information
  – Reduce sparseness
• Give seeds to several category learners
  – Seeds = Patterns or Datapoints
  – Add negative learners if possible
• Partition dataset
  – Relevant to some learner, or relevant to none
• For each learner:
  – Rank rules
  – Keep best
  – Rank datapoints
  – Keep best
• Repeat until convergence

Problem specification

• Depends on type of knowledge available
  – In particular, pre-processing
• Unconstrained search is controlled by
  – Modeling quality of rules and datapoints
  – Datapoints are judged on confidence, generality and number of rules
  – Dual judgement scheme for rules
• Convergence
  – Would like to know what conditions guarantee convergence

Co-training

• Key idea:
  – Disjoint views with ‘redundantly sufficient’ features
    – (Blum & Mitchell, 1999)
  – Simultaneously train two independent classifiers
    – Each classifier uses only one of the views
      – E.g. internal vs. external cues
• PAC-learnability results
  – Blum & Mitchell (1998)
  – Mitchell (1999)
**Co- and counter-training**

- Unsupervised learners help each other to bootstrap:
  - In co-training:
    - by providing reliable positive examples to each other
  - In counter-training:
    - by finding their own, weakly-reliable positive evidence
    - by providing reliable negative evidence to each other

- Unsupervised learners supervise each other

**Conclusions**

- Explored procedure for unsupervised acquisition of domain knowledge

- Respective merits of evaluation strategies

- *Multiple types of knowledge* essential for LT, as, for example, IE
  - Much more knowledge is needed for success in LT
  - Patterns $\rightarrow$ semantics (related to e.g., Barzilay 2001)
  - Names $\rightarrow$ synonyms/classes (e.g., Frantz et al.)

**Stevenson and Greenwood 2005**

*Semantic Approach to IE Pattern Induction*

**Outline**

- Approach to learning IE patterns which is an alternative to Yangarber et al.’s
  - Based on assumption that patterns with similar meanings are likely to be useful for extraction

**Learning Patterns**

Iterative Learning Algorithm

1. Begin with set of seed patterns which are known to be good extraction patterns
2. Compare every other pattern with the ones known to be good
3. Choose the highest scoring of these and add them to the set of good patterns
4. Stop if enough patterns have been learned, else goto 2.

**Semantic Approach**

- Assumption:
  - Relevant patterns are ones with similar meanings to those already identified as useful

- Example:
  - "The chairman resigned"
  - "The chairman stood down"
  - "The chairman quit"
  - "Mr. Smith quit the job of chairman"
Information Extraction and Weakly-supervised Learning

Patterns and Similarity

- Semantic patterns are SVO-tuples extracted from each clause in the sentence: chairman+resign
- Tuple fillers can be lexical items or semantic classes (e.g. COMPANY, PERSON)
- Patterns can be represented as vectors encoding the slot role and filler: chairman_subject, resign_verb
- Similarity between two patterns defined as follows:

\[
sim(\bar{a}, \bar{b}) = \frac{\bar{a} W \bar{b}^T}{\|\bar{a}\| \|\bar{b}\|}
\]

Matrix Population

- Matrix \( W \) is populated using semantic similarity metric based on WordNet
- \( W_{ij} = 0 \) for different roles or \( \text{sim}(w_i, w_j) \) using Jiang and Conrath’s (1997) WordNet similarity measure

<table>
<thead>
<tr>
<th></th>
<th>ceo_subject</th>
<th>resigned_verb</th>
<th>quit_verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>ceo+resigned</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>resigned+quit</td>
<td>0</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>quit+verb</td>
<td>0</td>
<td>0.9</td>
<td>1</td>
</tr>
</tbody>
</table>

- Semantic classes are manually mapped onto an appropriate WordNet synset

Advantage

- Adapted cosine metric allows synonymy and near-synonymy to be taken into account

Algorithm Setup

- At each iteration
  - each candidate pattern is compared against the centroid of the set of currently accepted patterns
  - patterns with score within 95% of best pattern are accepted, up to a maximum of 4
- Text pre-processed using GATE to tokenize, split into sentences and identify semantic classes
- Parsed using MINIPAR (adapted to deal with semantic classes marked in input)
- SVO tuples extracted from dependency tree

Evaluation

- MUC-6 “management succession” task

Example Learned Patterns

COMPANY+apoint+PERSON
PERSON+hire+PERSON
PERSON+succeed+PERSON
PERSON+appoint+PERSON
PERSON+name+POST
PERSON+join+COMPANY

PERSON+own+COMPANY
COMPANY+acquire+COMPANY
Comparison

• Compared with alternative approach
  "Document centric" method described by Yangarber, Grishman, Tapanainen and Huttunen (2000)
  Based on assumption that useful patterns will occur in documents similar to those which have already been identified as relevant

• Two evaluation regimes
  - Document filtering
  - Sentence filtering

Document Filtering Evaluation

• MUC-6 corpus (590 documents)
• Task involves identifying documents which contain management succession events
• Similar to MUC-6 document filtering task
• Document centric approach benefited from a supplementary corpus: 6,000 newswire stories from the Reuters corpus (3,000 with code “C411” = management succession events)

Document Filtering Results

Sentence Filtering Evaluation

• Version of MUC-6 corpus in which sentences containing events were marked (Soderland, 1999)
• Evaluate how accurately generated pattern set can distinguish between "relevant" (event describing) and non-relevant sentences

Sentence filtering results

Precision and Recall

• Semantic Similarity
• Document-centric
Error Analysis

- Event not described with SVO structure
  - Mr. Jones left Acme Inc.
  - Mr. Jones retired from Acme Inc.
- More expressive model needed

- Parse failure, approach depends upon accurate dependency parsing of input

Conclusion

- WordNet-based approach to weakly supervised pattern acquisition for Information Extraction
- Superior to prior approach on fine-grained evaluation
- Document filtering may not be best evaluation regime for this task

Outline

- Semantics
- Acquisition of semantic knowledge
  - Supervised vs unsupervised methods
  - Bootstrapping

Learning of Generalized Names

- On-line Demo: Incremental IFE-BIO database
  - Disease name
  - Location
  - Date
  - Victim number
  - Victim type/descriptor: people, animals, plants
  - Victim status: infected, sick, dead

- How do we get all these disease names
- COLING-2002: Yangarber, Lin & Grishman
**Motivation**

- For IE, we often need to identify names that refer to particular types of entities
- For IFE-BIO need names of:
  - Diseases
  - Agents
    - bacterium, virus, fungus, parasite, ...
  - Vectors
  - Drugs
  - ...
  - Locations

**Generalized names**

- Much prior work focuses on classifying proper names (PNs)
  - e.g. MUC Named Entity task (NE)
  - Person/Organization/Location
- For our purposes, need to identify and categorize generalized names (GNs)
  - Closer to Terminology: single- or multi-word domain-specific expressions
  - a different and more difficult task

**How GNs differ from PNs**

- Not necessarily capitalized:
  - tuberculosis
  - E. coli
  - Ebola hemorrhagic fever
  - variant Creutzfeldt–Jacob disease
- Name boundaries are non-trivial to identify:
  - “the four latest typhoid fever cases”
- Set of possible candidate names is broader and more difficult to determine
  - “National Veterinary Services Director Dr. Gideon Bruckner said no cases of foot and mouth disease have been found in South Africa…”
- Ambiguity
  - Shingles, AGE (acute gastro-enteritis), ...

**Why lists are “bad”**

- External, fixed lists are unsatisfactory:
  - Lists are never complete
  - all diseases, all villages
  - New names are constantly appearing
  - shifting borders
  - Humans perform with very high precision
- Alternative approach: learn names from context in a corpus
  - as humans do

**Algorithm Outline: Nomen**

- **Input**: Seed names in several categories
- Tag occurrences of names
- Generate local patterns around tags
- Match patterns elsewhere in corpus
  - Acquire top-scoring pattern(s)
- Acquired pattern tags new names
  - Acquire top-scoring name(s)
- Repeat

**Preprocessing**

- **Zoner**
  - Locate text-bearing zones:
    - Find story boundaries, strip mail headers, etc.
- **Tokenizer**
- **Lemmatizer**
- **POS tagger**
  - Some problems (distinguish active/passive):
    - mosquito-borne dengue
    - dengue-bearing mosquito
Seeds

- For each target category select N initial seeds:
  - diseases:
    - cholera, dengue, anthrax, BSE, rabies, JE, Japanese encephalitis, influenza, Nipah virus, FMD
  - locations:
    - United States, Malaysia, Australia, Belgium, China, Europe, Taiwan, Hong Kong, Singapore, France
  - others:
    - case, health, day, people, year, patient, death, number, report, farm
- Use N most common names
- Use additional categories

Pattern generation

- Tag every occurrence of each seed in corpus:
  - "... new cases of <dis> cholera <dis> this year in..."
- For each tag, (left and right) generate context rule:
  - [new case of <dis> cholera this year]
- Generalize candidate rules:
  - [new case of <dis> *   *   *   ]
  - [* case of <dis> *   *   *   ]
  - [* of <dis> *   *   *   ]
  - [* * * <dis> cholera this year]
  - [* * * <dis> cholera this *   ]
  - etc.
- Each rule predicts a left or right boundary

Pattern application

- Apply each potential pattern to corpus, observe where the pattern matches:
  - e.g., the rule [* * vacinate against <dis> * * * ]
- Each rule predicts one boundary: search for the partner boundary using a noun group regexp:
  - [Adj* Noun*]
  - "... distributed the yellow fever vaccine to the people"
- The resulting NG can be:
  - Positive: "... case of <dis> dengue <dis> ...
  - Negative: "... North of <loc> Malaysia <loc> ...
  - Unknown: "... symptoms of <?> swine fever <?> in...

Identify candidate NGs

- Sets of NGs that the pattern p matched
  - pos = distinct matched NG types of correct category
  - neg = distinct matched NG types of wrong category
  - unk = distinct matched NGs of unknown category
- Collect statistics for each pattern:
  - accuracy = pos/(pos + neg)
  - confidence = (pos - neg) / (pos + neg + unk)

Pattern selection

- Discard pattern p if acc(p) < θ
- The remaining rules are ranked by
  - Score(p) = conf(p) * log |pos(p)|
- Prefer patterns that:
  - Predict the correct category with less risk
  - Stronger support: match more distinct known names
- Choose top n patterns for each category: acquire
  - [* die of <dis> * * * ]
  - [* vaccinate against <dis> * * * ]
  - [* * <dis> outbreak that have]
  - [* * <dis> be endemic *]
  - [* case of <dis> * * * ]
Name selection

• Apply each accepted pattern to corpus, to find candidate names (using the noun group RE)
  - "More people die of profound heartbreak than grief."

• Rank each name type \( t \) based on quality of patterns that match it:
  \[
  \text{Rank}(t) = 1 - \prod_{p \in M_t} \left( 1 - \text{conf}(p) \right)
  \]
  - Require: \( | M_t | \geq 2 \Rightarrow t \) should appear \( \geq 2 \) times
  - more credit to types matched by more rules
  - \( \text{conf}(p) \) assigns more credit to reliable patterns

Parameters

• In experiments:
  - \( N = 10 \) (number of seeds)
  - \( \theta = 0.50 \) (accuracy threshold)
  - \( n = m = 5 \) (number of accepted patterns/types)

Related work

• Collins and Singer (1999)
  - Proper names (MUC NE-style)
  - person, organization, location
  - Full parse
  - Names must appear in certain restricted syntactic context
    - Apposition
    - Object of preposition (in a PP modifying a NP with a singular head)
  - Co-training: learn spelling and context separately
  - Accuracy 91.3%

• Riloff (1996) Riloff & Jones, 1999,
  - Riloff & al. (2002) x 2
    - Bootstrapping "semantic lexicons" using extraction patterns
    - Multiple categories:
      - Building, event, location, time, weapon, human
    - Recall "40–60%"
    - Precision 7

• Ciravegna (2001)
  - IE algorithm
  - Learn left and right boundaries separately
  - Multiple correction phases, to find most-likely consistent labeling
  - Supervised
    - CMU seminar announcements
    - Austin job ads
  - Use "mild" semantics
    - 89F
**Salient Features of Nomen**

- generalized names
- a few manually-selected seeds
- un-annotated corpus
- un-restricted contexts
- rules for left and right contexts independently
- multiple categories simultaneously

**Data**

- Articles from ProMed mailing list
- Full corpus:
  - 2.5 years: 100,000 sentences (5,100 articles)
  - Development corpus:
    - 6 months: 25,000 sentences (1,400 articles) 3.4Mb
- Realistic text
  - Written by medical professionals, only lightly edited
  - Variant spellings, misspellings
  - Other research (Frantzi, Ananiadou)
  - More challenging than newspaper text

**Automatic evaluation**

- Build three reference lists:
  - Manual: compiled from multiple external sources
  - Medical databases, web search, manual review...
  - Recall: appearing two or more times
  - Precision: add acronyms, strip generic heads

**Reference Lists**

- Make reference lists for each target category
- Score recall against the recall list, and precision against the precision list
- Categories:
  - Diseases
  - Locations
  - Symptoms
  - "Other" = negative category
- How many name types the algorithm learns correctly?

**Disease and Location Names**

**Evaluation of precision**

- Not possible to get full list of names for measuring precision
- Learns valid names not in our reference lists:
  - Diseases: rinderpest, konzo, Mediterranean spotted fever, coconut cadang-cadang, swamp fever, lathyrism, PRRS (for “porcine reproductive and respiratory syndrome”)
  - Locations: Kinta, Ulu Piah, Melilla, Anstohihy, …
- Precision is penalized unfairly
- Quantify this effect: add newly discovered names to precision list (only)
**Effect of understated precision**

- Found 99 new diseases: not found during manual compilation
- Encouraging result: algorithm fulfills its purpose

**Re-introduced names**

- Found 99 new diseases: not found during manual compilation
- Encouraging result: algorithm fulfills its purpose

**Competing categories**

- When learning too few categories, algorithm learns unselective patterns
  - "X has been confirmed"
- Too fine categorization may cause problems:
  - Metonymy may lower accuracy of good patterns
    - Inhibit learning
  - E.g., Agents vs. Diseases: “E. coli"
- Possible approach:
  - Learn metonymic classes together, as single category,
  - Then apply separate procedure to disambiguate

**Type-based vs. instance-based**

- Results not directly comparable to prior work
- Token/type dichotomy
- Token–based (instance–based):
  - Learner gets credit or penalty for each instance in corpus
- Type–based:
  - Learner gets credit once for each name, no matter how many times it appears in corpus
**Instance-based evaluation**

- More compatible with prior work
- Manually tag all instances of diseases and locations in a Test (sub-)corpus:
  - 500 sentences
  - (expert did not tag generics)
- Score same experiments, using MUC scoring tools (on each iteration)

**Token-based recall & precision**

- Larger training corpus yields increase in recall (with fixed test corpus)
- Contrast recall across 340 iterations
- Continue learning more rare types after #40

### MUC score vs corpus size

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Type-Based</th>
<th>Instance-Based</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.03</td>
<td>0.35</td>
</tr>
<tr>
<td>20</td>
<td>0.18</td>
<td>0.68</td>
</tr>
<tr>
<td>40</td>
<td>0.31</td>
<td>0.83</td>
</tr>
<tr>
<td>60</td>
<td>0.42</td>
<td>0.83</td>
</tr>
<tr>
<td>300</td>
<td>0.69</td>
<td>0.86</td>
</tr>
</tbody>
</table>

**Further improvements**

- Investigate more categories
  - vectors, agents, symptoms, drugs
- Different corpora and name categories
  - MUC, person/organization/location/artifact
- Extend noun group pattern for names
  - results shown are for [Adj* Noun+]
  - foot and mouth disease, legionnaires' disease
- Use finer generalization
  - POS
  - semantics

**Lin, Yangarber, Grishman 2003**

**Learning of Names and Semantic Classes in English and Chinese from Positive and Negative Examples**
Goals

- IE systems need to spot and classify names (or terms)
  - "There are reports of SARS from Ulu Piah."
- Unsupervised learning can help
  - Improve performance on disease/location task
  - Learn other categories
  - Multiple corpora
  - English and Chinese

Improvements

- More competing categories
  - symptom, animal, human, institution, time
- Refined noun group pattern
  - hyphens, apostrophes, location capitalization
- Revised criteria for best patterns and names

Named Entity Task

- Proper names: person, org, location
  - Use capitalization clues
- Hand-labeled evaluation set
  - MUC-7 training sets (150,000 words)
  - Token-based evaluation (MUC scorer)
- Training corpus:
  - Same authors as evaluation set
  - 3 million words

Type and Text Scores

Named Entities in Chinese

- Beijing University corpus
  - People's Daily, Jan. 1998 (700,000 words)
  - Manually word-segmented, POS-tagged, and NE-tagged
- Initial development environment:
  - Learn NEs, but rely on annotators’ segmentation and POS tags
- Re-tagged 41 documents (test corpus)
  - Native annotators omitted some organizations acronyms, and some generic terms
  - (produced enhanced-precision results)
**Proper names, no capitalization**

- Categories: person, org, location, other
- 50 seeds per category
- Hard to avoid generic terms
  - “department”, “committee”
  - Made a lexicon of common nouns that should not be tagged as names
  - Still penalized for multiword generics
    - “provincial government”

**Proper Names (Chinese)**

### Initial (seeds)

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>Chinese</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rec</td>
<td>Prec</td>
</tr>
<tr>
<td>Person type</td>
<td>3.1</td>
<td>98.3</td>
</tr>
<tr>
<td>text</td>
<td>2.2</td>
<td>71.2</td>
</tr>
<tr>
<td>Org type</td>
<td>5.3</td>
<td>100.0</td>
</tr>
<tr>
<td>text</td>
<td>5.8</td>
<td>97.6</td>
</tr>
<tr>
<td>Location type</td>
<td>20.1</td>
<td>94.2</td>
</tr>
<tr>
<td>text</td>
<td>19.7</td>
<td>92.5</td>
</tr>
</tbody>
</table>

### Final

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>Chinese</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rec</td>
<td>Prec</td>
</tr>
<tr>
<td>Person type</td>
<td>85.7</td>
<td>84.4</td>
</tr>
<tr>
<td>text</td>
<td>75.7</td>
<td>74.5</td>
</tr>
<tr>
<td>Org type</td>
<td>89.6</td>
<td>70.2</td>
</tr>
<tr>
<td>text</td>
<td>83.9</td>
<td>65.7</td>
</tr>
<tr>
<td>Location type</td>
<td>91.8</td>
<td>82.2</td>
</tr>
<tr>
<td>text</td>
<td>89.1</td>
<td>79.7</td>
</tr>
</tbody>
</table>
Introduction

Several of the systems we have looked at use extraction patterns consisting of SVO tuples extracted from dependency trees.


- SVO tuples are a pattern model
  - predefined portions of the dependency tree which can act as extraction patterns

- Sudo et. al. (2003) compares three different IE pattern models:
  1. SVO tuples
  2. The chain model
  3. The subtree model

Predicate Argument Model

Pattern consists of a subject-verb-object tuple; Yangarber (2003); Stevenson and Greenwood (2005)

Chain Model

Extraction patterns are chain-shaped paths in the dependency tree rooted at a verb; Sudo et. al. (2001), Sudo et. al. (2003)

Subtree Model

Patterns are any subtree of the dependency tree consisting of at least two nodes

By definition, contains all the patterns proposed by the previous models; Sudo et. al. (2003)
**Pattern Relations**

- **Subtrees**
- **SVO**
- **Chains**

**Experiment**

- The task was to identify all the entities participating in events from two sets of Japanese texts.
  1. **Management Succession scenario**: Person, Organisation and Post
  2. **Murder/Arrest scenario**: Suspect, Arresting agency, Charge
- Does not involve grouping entities involved in the same event.
- Patterns for each model were generated and then ranked (ordered)
  - A pattern must contain at least one named entity class

**Ranking Subtree Patterns**

- Ranking of subtree patterns inspired by TF/IDF scoring.
  - Term frequency, $tf_i$ – the raw frequency of a pattern
  - Doc frequency, $df_i$ – the number of docs in which a pattern appears
  - Ranking function, score, is then:

$$score = tf_i \left( \log \frac{N}{df_i} \right)^\beta$$

**Management Succession Results**

**Discussion**

- **Advantages of Subtree model**:
  - Allows the capture of more varied context
  - Can capture more scenario specific patterns
- **Disadvantages of the Subtree model**:
  - Added complexity of many more patterns to process
  - Not clear that results are significantly better than predicate–argument or chain models.
**Linked Chain Model**
- A new pattern model introduced by Greenwood et. al. (2005)
- Patterns are chains or any pair of chains sharing their root

```
- hire/V
- nsubj
- IBM/N
- nobj
- Smith/N
- resign/V
- nsubj
- after
- nsubj
- IBM/N
- nobj
- Jones/N
```

**Pattern Relations**

**Choosing an Appropriate Pattern Model**
- An appropriate pattern model should balance two factors:
  - **Expressivity**: the model needs to be able to represent the items to be extracted from text
  - **Simplicity**: the model should be no more complex than it needs to be

**Pattern Enumeration**
- Choice of model affects the number of possible extraction patterns

```
Model Patterns
SVO 3
Chains 18
Linked Chains 66
Subtree 245
```

\[
N_{SV0}(T) = |V|
\]

Now let \(d(v)\) be the number of nodes obtained by taking a node \(v\), a member of \(V\), and all its descendants.

\[
N_{chains}(T) = \sum_{v \in V} (d(v) - 1)
\]

\[
N_{subtree}(T) = \sum_{n \in N} \frac{1}{|v_{chains}(n)|}
\]

- The number of subtrees can be defined recursively:

\[
N_{subtree}(T) = \left( \sum_{n \in N} \frac{1}{|v_{chains}(n)|} \right) - |N|
\]
**Pattern Expressiveness**

- The models include different parts of a sentence.
  - “Smith joined Acme Inc. as CEO”

  ![Pattern Tree Diagram]

  - SVO: “Smith” – “Acme”
  - Chains: “Acme” – “CEO”
  - Linked chains and subtrees: both

**Experiments**

- Aim to identify how well each pattern model captures the relations occurring in an IE corpus
- Extract patterns from a parsed corpus and, for each model, check whether it contains the related items
- Two corpora were used:
  1. MUC6 management succession texts
  2. Corpora of biomedical text

**Management Succession Corpus**

- Stevens succeeds Fred Casey who retired from the OCC in June
- PersonIn: “Stevens”
- PersonOut: “Fred Casey”
- Company: “OCC”

**Biomedical Corpus**

- Combination of three corpora, each containing binary relations
- Gene–protein interactions
  - Expression of $\text{Sigma(K)}$-dependent $\text{cwlH}$ gene depended on $\text{perE}$
- Relations between genes and diseases
  - Most sporadic colorectal cancers also have two $\text{APC}$ mutations

**Parsers**

1. MINIPAR (Lin, 1999)
5. RASP (Briscoe and Carroll, 2002)

**Pattern Counts**

<table>
<thead>
<tr>
<th>Model</th>
<th>SVO</th>
<th>Chains</th>
<th>Linked Chains</th>
<th>Subtrees</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minipar</td>
<td>2,980</td>
<td>52,659</td>
<td>149,504</td>
<td>$1.40 \times 10^4$</td>
</tr>
<tr>
<td>Machine Syntax</td>
<td>2,382</td>
<td>67,600</td>
<td>265,631</td>
<td>$4.64 \times 10^3$</td>
</tr>
<tr>
<td>Stanford</td>
<td>2,950</td>
<td>76,620</td>
<td>478,643</td>
<td>$1.69 \times 10^3$</td>
</tr>
<tr>
<td>Malt</td>
<td>2,061</td>
<td>90,587</td>
<td>697,223</td>
<td>$4.55 \times 10^3$</td>
</tr>
<tr>
<td>RASP</td>
<td>2,930</td>
<td>70,804</td>
<td>250,806</td>
<td>$5.73 \times 10^3$</td>
</tr>
</tbody>
</table>
**Evaluating Expressivity**

- A pattern covers a relation if it includes both related items.
- The expressivity of each model is measured in terms of the percentage of relations which are covered by each pattern.

\[
\text{coverage} = \frac{\# \text{ of relations covered by model}}{\# \text{ of relations in corpus}}
\]

- Not an extraction task!

**Result Summary**

- Average coverage for each pattern model over all texts

**Analysis**

- Differences between models is significant (one way repeated measures ANOVA, p < 0.01)
- Tukey test revealed no significant differences (p < 0.01) between
  - Linked chains and subtree
  - SVO and chains

**Fragmentation and Coverage**

- Strong negative correlation (r = -0.92) between average number of fragments produced by a parser and coverage of the subtree model
- Not very surprising but suggests a very simple way to decide between parsers

**Bounded Coverage**

- Analysis showed that parsers often failed to generate a spanning parse
- None of the models can perform better than the subtree model
- Results for the SVO, chain and linked chain models can be interpreted in terms of the percentage of relations which were identified by the subtree model

\[
\text{bounded coverage} = \frac{\# \text{ of relations covered by model}}{\# \text{ of relations covered by subtree model}}
\]

**Management Succession Results**

- SVO and chains do not cover many of the relations
- Subtree and linked chains models have roughly same coverage
**Biomedical Results**

<table>
<thead>
<tr>
<th>Parser</th>
<th>SVO</th>
<th>Chains</th>
<th>Linked Chains</th>
<th>Subtrees</th>
</tr>
</thead>
<tbody>
<tr>
<td>MINIPAR</td>
<td>0.93%</td>
<td>41%</td>
<td>65% (92%)</td>
<td>71%</td>
</tr>
<tr>
<td>Machinese Syntax</td>
<td>0.19%</td>
<td>10%</td>
<td>3% (9%)</td>
<td>2%</td>
</tr>
<tr>
<td>Stanford</td>
<td>0.46%</td>
<td>13%</td>
<td>89% (93%)</td>
<td>95%</td>
</tr>
<tr>
<td>Malt</td>
<td>0.23%</td>
<td>10%</td>
<td>73% (82%)</td>
<td>87%</td>
</tr>
<tr>
<td>RASP</td>
<td>0.5%</td>
<td>7%</td>
<td>39% (85%)</td>
<td>47%</td>
</tr>
</tbody>
</table>

- SVO covers very few of the relations
- Bounded coverage for all models is lower than management succession domain

**Individual Relations**

- Bounded coverage for each relation and model combination using Stanford parser

<table>
<thead>
<tr>
<th>Relationship</th>
<th>SVO</th>
<th>Chains</th>
<th>Linked Chains</th>
</tr>
</thead>
<tbody>
<tr>
<td>PersonOut-Company</td>
<td>2.69%</td>
<td>40.86%</td>
<td>91.40%</td>
</tr>
<tr>
<td>PersonIn-Company</td>
<td>5.30%</td>
<td>18.94%</td>
<td>95.45%</td>
</tr>
<tr>
<td>PersonOut-Post</td>
<td>14.71%</td>
<td>58.89%</td>
<td>95.80%</td>
</tr>
<tr>
<td>PersonIn-Post</td>
<td>34.42%</td>
<td>25.32%</td>
<td>97.73%</td>
</tr>
<tr>
<td>PersonIn-PersonOut</td>
<td>23.30%</td>
<td>17.65%</td>
<td>96.08%</td>
</tr>
<tr>
<td>Post-Company</td>
<td>5.10%</td>
<td>58.37%</td>
<td>93.00%</td>
</tr>
<tr>
<td>Gene-Disease</td>
<td>96.08%</td>
<td>17.65%</td>
<td>92.72%</td>
</tr>
<tr>
<td>PersonOut-PersonOut</td>
<td>0.00%</td>
<td>25.72%</td>
<td>90.04%</td>
</tr>
</tbody>
</table>

- SVO does better than chains for two relations: PersonIn-Post and PersonIn-PersonOut
  - Often expressed using simple predicate-argument structures:
    - “PersonIn succeeds PersonOut”
    - “PersonOut will be succeeded by PersonIn”
    - “PersonIn will become Post”
    - “PersonIn was named Post”

- Chains do best on four relations
  - PersonOut-Company and PersonOut-Post: appositions or relative clauses

  “PersonOut, a former CEO of Company,”
  “current acting Post, PersonOut,”
  “PersonOut, who was Post,”
  “PersonOut, who is still Post,”

- Gene–Disease
  - “Gene, the candidate gene for Disease,”
  - “the gene for Disease, Gene,”

- Post–Company
  - prepositional phrase or possessive
    - “Post of Company”
    - “Company’s Post”

**Linked Chains**

- Examples covered by linked chains but not SVO or chains usually expressed within a predicate–argument structure in which the related items are not the subject and object
  - “Company announced a new CEO, PersonIn”
**Information Extraction and Weakly-supervised Learning**

**Pattern Comparison**
- Repeat of Sudo et. al.’s pattern ranking experiment
- \[ \text{score} = tf \left( \log \frac{N}{df} \right)^n \]
- Four pattern models compared
- Extraction task taken from MUC–6

**Pattern Generation**
- Patterns generated for each model
  - Efficient algorithm was unable to generate all subtrees (Abe et. al. 2002; Zaki 2002)
- Generated two sets of patterns:
  - Filtered: occurred at least four times
  - Unfiltered: SVO, chain and linked chain only

<table>
<thead>
<tr>
<th>Model</th>
<th>Filtered</th>
<th>Unfiltered</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVO</td>
<td>61,88</td>
<td>23,128</td>
</tr>
<tr>
<td>Chains</td>
<td>18,164</td>
<td>142,018</td>
</tr>
<tr>
<td>Linked chains</td>
<td>23,452</td>
<td>493,463</td>
</tr>
<tr>
<td>Subtrees</td>
<td>369,453</td>
<td>1.69 \times 10^6</td>
</tr>
</tbody>
</table>

Unfiltered subtrees not generated

**Results: Filtered Patterns**

**Discussion**
- Linked chain and subtree models have similar performance
- Chain model performs poorly
- Three highest ranked SVO patterns have extremely high precision
  - PERSON–succeed–PERSON (P = 90.1%)
  - PERSON–be–POST (P = 80.8%)
  - PERSON–become–POST (P = 78.9%)
- (If these patterns were removed the maximum SVO precision would be 32%)
Results: Unfiltered Pattern

- Filtered subtrees included for comparison

Discussion

- Extra patterns for SVO, chain and linked chain models improve recall without affecting the maximum precision for each model
- Linked chain model benefits more than SVO or chain model and achieves for higher recall than other models
  - Only model which is able to represent relation in this corpus

Summary

- Comparison of four models for Information Extraction patterns based on dependency trees
- Linked chain model represents a good balance between pattern complexity and tractability
  - But have problems with certain linguistic constructions