CS674 NLP

Information Extraction

Claire Cardie
Cornell University

Information extraction

- **Acquiring extraction patterns**
  - Learning approaches
    - Semi-automatic methods for extraction from unstructured text
    - Fully automatic methods for extraction from structured text
  - Finite-state methods

IE system: input

SAN SALVADOR, 15 JAN 90 (ACAN-EFE) -- [TEXT] ARMANDO CALDERON SOL, PRESIDENT OF THE NATIONALIST REPUBLICAN ALLIANCE (ARENA), THE RULING SALVADORAN PARTY, TODAY CALLED FOR AN INVESTIGATION INTO ANY POSSIBLE CONNECTION BETWEEN THE MILITARY PERSONNEL IMPLICATED IN THE ASSASSINATION OF JESUIT PRIESTS.

"IT IS SOMETHING SO HORRENDOUS, SO MONSTROUS, THAT WE MUST INVESTIGATE THE POSSIBILITY THAT THE FMLN (FARABUNDO MARTI NATIONAL LIBERATION FRONT) STAGED THIS ASSASSINATION TO DISCREDIT THE GOVERNMENT," CALDERON SOL SAID.

SALVADORAN PRESIDENT ALFREDO CRISTIANI IMPLICATED FOUR OFFICERS, INCLUDING ONE COLONEL, AND FIVE MEMBERS OF THE ARMED FORCES IN THE ASSASSINATION OF SIX JESUIT PRIESTS AND TWO WOMEN ON 16 NOVEMBER AT THE CENTRAL AMERICAN UNIVERSITY.

IE system: output

1. DATE	- 15 JAN 90
2. LOCATION	EL SALVADOR: CENTRAL AMERICAN UNIVERSITY
3. TYPE	MURDER
4. STAGE OF EXECUTION	ACCOMPLISHED
5. INCIDENT CATEGORY	TERRORIST ACT
6. PERP: INDIVIDUAL ID	"FOUR OFFICERS" "ONE COLONEL" "FIVE MEMBERS OF THE ARMED FORCES"
7. PERP: ORGANIZATION ID	"ARMED FORCES", "FMLN"
8. PERP: CONFIDENCE	REPORTED AS FACT
9. HUM TGT: DESCRIPTION	"JESUIT PRIESTS" "WOMEN"
10. HUM TGT: TYPE	CIVILIAN: "JESUIT PRIESTS" CIVILIAN: "WOMEN"
11. HUM TGT: NUMBER	6: "JESUIT PRIESTS" 2: "WOMEN"
12. EFFECT OF INCIDENT	DEATH: "JESUIT PRIESTS" DEATH: "WOMEN"

Issues for learning extraction patterns

- **Training data is difficult to obtain**
  - IE answer keys provide some supervisory information --- string to be extracted and its label --- but often not enough
  - No direct means for learning "set fills"
  - Training examples usually encode the output of earlier levels of syntactic and semantic analysis
    - No standard training set available
    - When these "preprocessing" components change, examples must be regenerated
  - Standard "off-the-shelf" learning algorithms tend to work less well than those specifically tailored to the task

Learning IE patterns from examples

- **Goal**
  - Given a training set of documents paired with human-produced filled extraction templates [answer keys],
  - Learn extraction patterns for each slot using an appropriate machine learning algorithm.
- **Options**
  - Memorize the fillers of each slot
  - Generalize the fillers using
    - p-o-s tags?
    - phrase structure (NP, V) and grammatical roles (SUBJ, OBJ)?
    - semantic categories?

Learning IE patterns

- **Methods vary with respect to**
  - The model class of pattern learned (e.g. lexically based regular expression, syntactic-semantic pattern)
  - Training corpus requirements
  - Amount and type of human feedback required
  - Degree of pre-processing necessary
  - Background knowledge presumed

Autoslog [Riloff 1993]

- **Learns syntactico-semantic patterns (called "concept nodes")**

Sentence Two: "Witnesses confirm that the twister occurred without warning at approximately 7:15 p.m and *destroyed two mobile homes.*"

Concept Node Definition:
Concept = Damaged-Object
Trigger = "destroyed"
Position = direct-object
Constraints = ((physical-object))
Enabling Conditions = ((active-voice))

Instantiated Concept Node
Damaged-Object = "two mobile homes"

Figure 3: Concept Node for Extracting "Damage" Information.

from Cardie [1997]

Autoslog algorithm

- **Noun phrase extraction only**
- **Relies on a small set of pattern templates**
  - <active-voice-verb> <direct object>=<target-np>
  - <subject>=<target-np> <active-voice-verb>
  - <subject>=<target-np> <passive-voice-verb>
  - <passive-voice-verb> by <object>=<target-np>
  - …
  - – Domain-independent
  - – So require little modification when switching domains
- **Requires partial parser**
- **Assumes semantic category(ies) for each slot are known, and all potential slot fillers can be tested w.r.t. them**

Autoslog algorithm

- **Find the sentence from which the noun phrase originated.**
- **Present the sentence to the partial parser.**
- **Apply the pattern templates in order.**
- **When a pattern applies, generate a concept node definition from the matched constituent, its context, the slot type (from the answer key), and the (predefined) semantic class for the filler.**

Concept = < <concept> of <target-np> >
Trigger = "< <verb> of <active-voice-verb> >"
Position = direct-object
Constraints = ((< <semantic class> of <concept> >))
Enabling Conditions = ((active-voice))

Learned terrorism patterns

- **<victim> was murdered**
- **<perpetrator> bombed**
- **<perpetrator> attempted to kill**
- **was aimed at <target>**

Natural disasters patterns

<subject> = disaster-event (earthquake) registered (active)
registered (active) <direct obj> = magnitude

Yesterday's earthquake registered 6.9 on the Richter scale.

measuring (gerund) <direct obj> = magnitude

measuring 6.9 …

aid (noun)…in/to/for (prep) <obj> = disaster-event-location/ victim

…sending medical aid to Afghanistan…
…sending medical aid to earthquake victims…

Advantages and Disadvantages

- **Learns bad patterns as well as good patterns**
  - Too general (e.g. triggered by “is” or “are” or by verbs not tied to the domain)
  - Too specific
  - Just plain wrong
    - Parsing errors
    - Target NPs occur in a prepositional phrase and Autoslog can’t determine the trigger (e.g. is it the preceding verb or the preceding NP?)
- **Requires that a person review the proposed extraction patterns, discarding bad ones**
- **No computational linguist needed (?)**
- **Reduced human effort from 1200-1500 hours to ~4.5 hours**
- **F-measure dropped from 50.5 to 48.7 (for one test set); from 41.9 to 41.8 (for a second test set)**

Autoslog-TS

- **Largely unsupervised**
- **Two sets of documents: relevant, not relevant**
- **Apply pattern templates to extract every NP in the texts**
- **Compute *relevance rate* for each pattern *i* :**

Pr (relevant text | text contains i) =
freq of *i* in relevant texts / frequency of *i* in corpus

- **Sort patterns according to relevance rate and frequency**

relevance rate * log (freq)

Autoslog-TS

- Human review of learned patterns required
- Also requires labeling the semantic category of the extracted slot filler

Learning extraction patterns

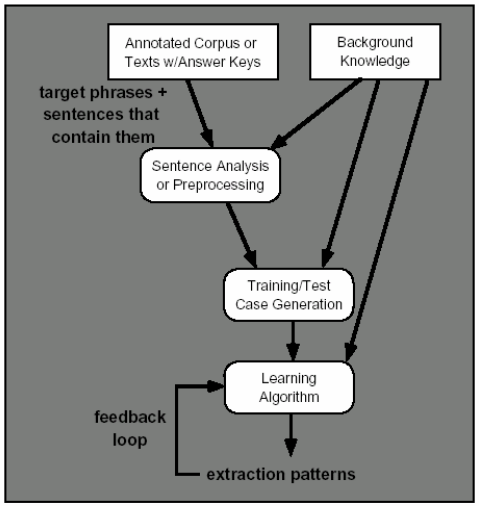

Figure 4: Learning Information Extraction Patterns.

Information extraction

- **Acquiring extraction patterns**
  - Learning approaches
    - Semi-automatic methods for extraction from unstructured text
    - Fully automatic methods for extraction from structured text
  - Finite-state methods

Covering algorithms

- **E.g. Crystal** [Soderland et al. 1995]
  - Allows for more complicated patterns
    - Can test target NP or any constituent in its context for
      - presence of any word or sequence of words
      - semantic class of heads or modifiers
- **Covering algorithm: successively generalizes the extraction patterns until the generalization produces errors**
  - Generate the most specific pattern possible for every phrase to be extracted in the training corpus
  - For each pattern, P, find the most similar pattern P' and relax the constraints of each just enough to unify P and P'.
  - Test the new extraction pattern E against the training corpus.
  - If its error rate is < threshold T, add E to the set of patterns, replacing P and P'.
  - Repeat the process on E until the error tolerance is exceeded.
  - Move on to the next pattern, P, in the original set

Extraction patterns for semi-structured text

- **If extracting from automatically generated web pages, simple regex patterns usually work.**
- **Specify an item to extract for a slot using a regular expression pattern.**
  - Price pattern: "\b\$\d+(\.\d{2})?\b"
- **May require preceding (pre-filler) pattern to identify proper context.**
  - Amazon list price:
    - Pre-filler pattern: "List Price: "
    - Filler pattern: "\$\d+(\.\d{2})?\b"
- **May require succeeding (post-filler) pattern to identify the end of the filler.**
  - Amazon list price:
    - Pre-filler pattern: "List Price: "
    - Filler pattern: ".+"
    - Post-filler pattern: ""

Simple template extraction

- **Extract slots in order, starting the search for the filler of the *n*+1 slot where the filler for the *n*th slot ended. Assumes slots always in a fixed order.**
  - Title
  - Author
  - List price
  - …
- **Make patterns specific enough to identify each filler always starting from the beginning of the document.**
- **Rapier system learns three regex-style patterns for each slot: pre-filler, filler, post-filler**

Extraction patterns for semi-structured text

- **If extracting from more natural, unstructured, human-written text, some NLP will usually help.**
  - Part-of-speech (POS) tagging
    - Mark each word as a noun, verb, preposition, etc.
  - Syntactic parsing
    - Identify phrases: NP, VP, PP
  - Semantic word categories (e.g. from WordNet)
    - KILL: kill, murder, assassinate, strangle, suffocate
  - E.g. Rapier's extraction patterns can use POS or phrase tags.
    - Crime victim:
      - Prefiller: [POS: V, Hypernym: KILL]
      - Filler: [Phrase: NP]

Set fill extraction

- **If a slot has a fixed set of pre-specified possible fillers, text categorization can be used to fill the slot.**
  - Job category
  - Company type
- **Treat each of the possible values of the slot as a category, and classify the entire document to determine the correct filler.**
- **When won't this work?**

XML and IE

- **If relevant documents were all available in standardized XML format, IE would be unnecessary.**
- **But…**
  - Difficult to develop a universally adopted DTD format for the relevant domain.
  - Difficult to manually annotate documents with appropriate XML tags.
  - Commercial industry may be reluctant to provide data in easily accessible XML format.
- **IE provides a way of automatically transforming semi-structured or unstructured data into an XML compatible format.**