Information Retrieval

Lecture 6: Information Extraction and Bootstrapping

Computer Science Tripos Part II

UNIVERSITY OF CAMBRIDGE

Simone Teufel

Natural Language and Information Processing (NLIP) Group

sht25@cl.cam.ac.uk

Last time

- Range of problems that make named entity recognition (NE) hard
- Mikheev et al's (1998) cascading NE system
- NE is the simplest kind of IE task: no relations between entities must be determined
- NIST MUC conferences pose three kinds of harder IE tasks
- Today: more of the full task (scenario templates), and on learning

Lexico-semantic patterns

- “Flattened-out” semantic representations with lexemes directly hard-wired into them
- String-based matching with type of semantic category to be found directly expressed in lexical pattern
- Problem with all string-based mechanisms: generalisation to other strings with similar semantics, and to only those
- Do generalisation by hand...
  - <Perpetrator> (APPOSITION) {blows/blew/has blown} {himself/herself} up
  - <Perpetrator> detonates
  - {blown up/detonated} by <Perpetrator>
- Manual production of patterns is time-consuming, brittle, and not portable across domains

Learning of lexico-semantic patterns (Riloff 1993)

- UMASS participant system in MUC-4: AutoSlog
- Lexico-semantic patterns for MUC-3 took 1500 person hours to build $\rightarrow$ knowledge engineering bottleneck
- AutoSlog achieved 98% performance of manual system; AutoSlog dictionary took 5 person hours to build
- “Template mining:”
  - Use MUC training corpus (1500 texts + human answer keys; 50% non-relevant texts) to learn contexts
  - Have human check the resulting templates (30% - 70% retained)

- 389 Patterns ("concept nodes") with enabling syntactic conditions, e.g. active or passive:
  - kidnap-passive: <VICTIM> expected to be subject
  - kidnap-active: <PERPETRATOR> expected to be subject
- Hard and soft constraints for fillers of slots
  - Hard constraints: selectional restrictions; soft constraints: semantic preferences
- Semantic lexicon with 5436 entries (including semantic features)

Heuristics for supervised template mining (Riloff 1993)

- Stylistic conventions: relationship between entity and event made explicit in first reference to the entity
- Find key word there which triggers the pattern: *kidnap, shot,*
- Heuristics to find these trigger words
- Given: filled template plus raw text. Algorithm:
  - Find first sentence that contains slot filler
  - Suggest good conceptual anchor point (trigger word)
  - Suggest a set of enabling conditions

“the diplomat was kidnapped” + VICTIM: the diplomat

Suggest: <SUBJECT> passive-verb + trigger=kidnap

Learning of lexico-semantic patterns (Riloff 1993)

System uses 13 heuristics:

<victim> was murdered	(<subject>, passive-verb)
<perpetrator> bombed	(<subject>, active-verb)
<perpetrator> attempted to kill	(<subject> verb infinitive)
<victim> was victim	(subject auxiliary <noun>)
killed <victim>	(passive-verb <dobj>)
bombed <target>	(active-verb <dobj>)
to kill <victim>	(infinitive <dobj>)
threatened to attack <target>	(verb infinitive <dobj>)
killing <victim>	(gerund <dobj>)
fatality was <victim>	(noun auxiliary <dobj>)
bomb against <target>	noun prep <np>
killed with <instrument>	active-verb prep <np>
was aimed at <target>	passive-verb prep <np>

Riloff 1993: a good concept node

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 (*S* 1))
Constraints:	(class phys-target *S*)
Constant slots:	(type bombing)
Enabling Conditions:	((passive))

Riloff 1993: another good concept node

ID: DEV-MUC4-0071
Slot Filler: “guerrillas
Sentence: THE SALVADORAN GUERRILLAS ON MAR_12_89, TODAY, THREATENED TO MURDER INDIVIDUALS INVOLVED IN THE MAR_19_88 PRESIDENTIAL ELECTIONS IF THEY DO NOT RESIGN FROM THEIR POSTS.

CONCEPT NODE	
Name:	perpetrator-subject-verb-infinitive-threatened-to-murder
Trigger:	murder
Variable slots:	(perpetrator (*S* 1))
Constraints:	(class perpetrator *S*)
Constant slots:	(type perpetrator)
Enabling Conditions:	((active) (trigger-preceded-by? ’to ’threatened))

Riloff 1993: a bad concept node

ID: DEV-MUC4-1192
Slot Filler: “gilberto molasco
Sentence: THEY TOOK 2-YEAR-OLD GILBERTO MOLASCO, SON OF PATRICIO RODRIGUEZ, AND 17-OLD ANDRES ALGUETA, SON OF EMIMESTO ARGUETA.

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

Riloff 1993: evaluation

System/Test Set	Recall	Prec	F-measure
MUC-4/TST3	46	56	50.5
AutoSlog/TST3	43	56	48.7
MUC-4/TST4	44	40	41.9
AutoSlog/TST4	39	45	41.8

- 5 hours of sifting through AutoSlog's patterns
- Porting to new domain in less than 10 hours of human interaction
- But: creation of training corpus ignored in this calculation

Agichtein, Gravano (2000): Snowball

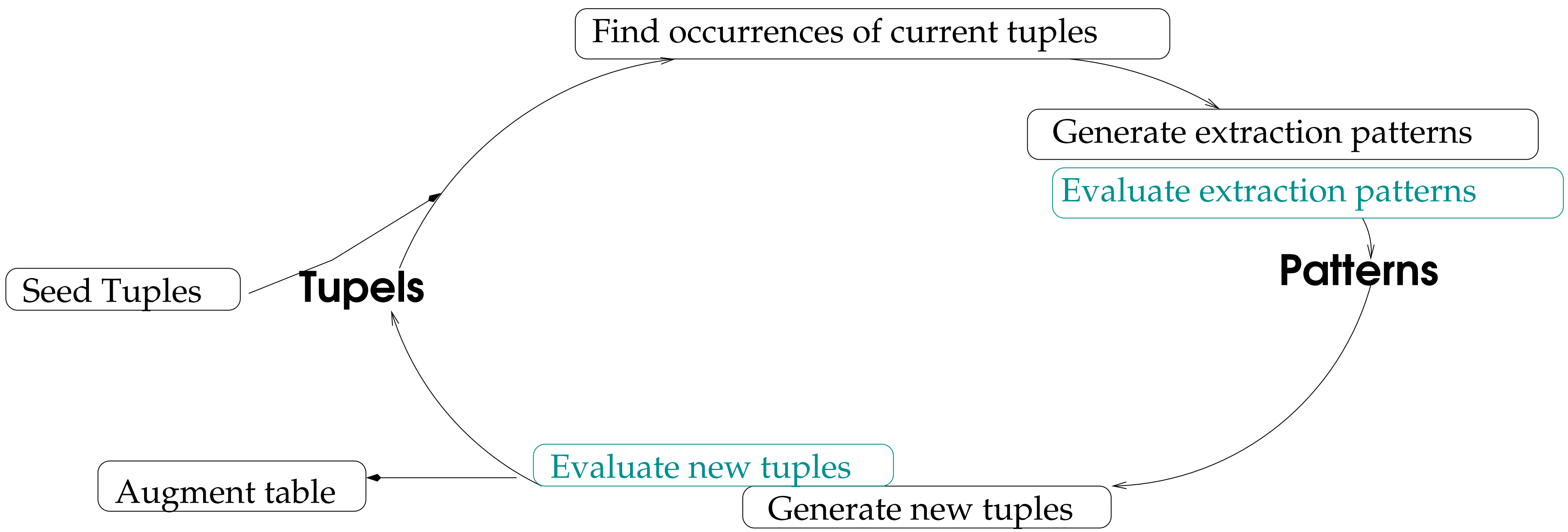
- Find locations of headquarters of a company and the corresponding company name ($< o, l >$ tuples)

Organisation	Location of Headquarters
Microsoft	Redmond
Exxon	Irving
IBM	Armonk
Boeing	Seattle
Intel	Santa Clara

  "Computer servers at Microsoft's headquarters in Redmond"

- Use minimal human interaction (handful of positive examples)
  - no manually crafted patterns
  - no large annotated corpus (IMass system at MUC-6)
- Automatically learn extraction patterns
- Less important to find every occurrence of patterns; only need to fill table with confidence

Agichtein, Gravano (2000): Bootstrapping



Agichtein, Gravano (2000): Overall process

- Start from table containing some $< o, l >$ tuples (which must exist in document collection)
- Perform NE (advantage over prior system DIPRE (Brin 98))
- System searches for occurrences of the example $< o, l >$ tuples in documents
- System learns extraction patterns from these example contexts, e.g.:

<ORGANIZATION> 's headquarters in <LOCATION>
<LOCATION>-based <ORGANIZATION>

- Evaluate patterns; use best ones to find new $< o, l >$ tuples
- Evaluate new tuples, choose most reliable ones as new seed tuples
- Iteratively repeat the process

Agichtein, Gravano (2000): Context generalisation and patterns

A SNOWBALL pattern is a 5-tuple <left,tag1,middle,tag2,right>

left	Tag1	middle	Tag2	right
The	Irving	-based	Exxon Corporation	
<{<the, 0.2>},	LOCATION,	{<-,0.5> <based, 0.5>},	ORGANIZATION,	{} >

- Associate term weights as a function of frequency of term in context
- Normalize each vector so that norm is 1; then multipy with weights W_left, W_right, W_mid.
- Degree of match between two patterns $t_p =< l_p, t_1, m_p, t_2, r_p >$ and $t_s =< l_s, t'_1, m_s, t'_2, r_s >$:

$$match(t_p, t_s) = l_p l_s + m_p m_s + r_p r_s \text{ (if tags match, 0 otherwise)}$$

Agichtein, Gravano (2000): Pattern generation

- Similar contexts form a pattern
  - Cluster vectors using a clustering algorithm (minimum similarity threshold τ_{sim})
  - Vectors represented as cluster centroids $\bar{l_s}, \bar{m_s}, \bar{r_s}$
- Generalised Snowball pattern defined via centroids:

$$< \bar{l_s}, tag_1, \bar{m_s}, tag_2, \bar{r_s} >$$

- Remember for each Generalised Snowball pattern
  - All contexts it came from
  - The distances of contexts from centroid

Agichtein, Gravano (2000): Productivity/Reliability

- We want productive and reliable patterns (and tuples produced by these)
  - productive but not reliable:
    $< \{\}, ORGANIZATION, \{<'','', 1 >\}, LOCATION, \{\} >$

    "Intel, Santa Clara, announced that. . . "

    "Invest in Microsoft, New York-based analyst Jane Smith said. . . "
  - reliable but not productive:
    $< \{\}, ORGANIZATION, \{< whose, 0.1 >, < headquarter, 0.4 >, < is, 0.1 >< located, 0.3 >, < in, 0.09 >, < nearby, 0.01 >\}, LOCATION, \{\} >$

    "Exxon, whose headquarter is located in nearby Irving. . . "
- Eliminate patterns supported by less than $\tau_{sup} < o, l >$ tuples

Agichtein, Gravano (2000): Pattern reliability

$P = < \{\}, ORGANIZATION, \{<'',''\ ,1>\}, LOCATION, \{\} >$

- Pattern P matches in three contexts (returns three tuples):
  - Exxon, Irving, said
  - Intel, Santa Clara, cut prices
  - invest in Microsoft, New York-based analyst Jane Smith said
- We know that
  - $<$Exxon, Irving$>$ and $<$Intel, Santa Clara$>$ are correct
  - $<$Microsoft, New York$>$ cannot be correct (as $<$Microsoft, Redmont$>$ is in our table)
- If P predicts tuple $t = < o, l >$ and there is already tuple $t' = < o, l' >$ with high confidence, then: if $l = l' \rightarrow P.positive$++, otherwise $P.negative$++ (uniqueness constraints: organization is key)
- $Conf(P) = \frac{P.positive}{P.positive+P.negative} = \frac{2}{2+1}$ (range [0..1])

Agichtein, Gravano (2000): Pattern confidence

- Consider productivity, not just reliability:

$$Conf_{RlogF}(P) = Conf(P)log_2(P.positive)$$

- Normalized $Conf_{RlogFNorm}(P)$:

$$Conf_{RlogFNorm}(P) = \frac{Conf_{RlogF}(P)}{max_{i \in \mathcal{P}} Conf(i)}$$

  (this brings $Conf_{RlogFNorm}(P)$ into range [0...1])

- $max_{i \in \mathcal{P}} Conf(i)$ is the largest confidence value seen with any pattern

- $Conf_{RlogFNorm}(P)$ is a rough estimate of the probability of pattern P producing a valid tuple

- Confidence of a tuple T is probability that at least one valid tuple is produced:

$$Conf(T) = 1 - \prod_{i=0}^{|P|} (1 - Conf(P_i) Match(C_i, P_i))$$

- Reason: probability of every pattern matched incorrectly:

$$Prob(T \text{ is NOT valid}) = \prod_{i=0}^{|P|} (1 - P(i))$$

$P = \{P_i\}$ is the set of patterns that generated T

C_i is the context associated with an occurrence of T

$Match(C_i, P_i)$ is goodness of match between P_i and C_i

- Formula due to the assumption that for an extracted tuple T to be valid, it is sufficient that at least one pattern matched the “correct” text context of T.

Agichtein, Gravano (2000): Tuple evaluation II

- Then reset confidence of patterns:

$$Conf(P) = Conf_{new}(P)W_{updt} + Conf_{old}(P)(1 - W_{updt})$$

  W_{updt} controls learning rate: does system trust old or new occurrences more? Here: $W_{updt} = 0.5$

- Throw away tuples with confidence $< \tau_t$

Agichtein, Gravano (2000): Results

Conf	middle	right
1	<based, .53>, <in, .53>	<”,” ,.01>
.69	<”””, .42>,<s, .42>,<headquarters, .42>,<in,.42>	
.61	<(,.93>	<),.12>

- Use training corpus to set parameters: τ_{sim}, τ_t, τ_{sup}, I_{max}, W_{left}, W_{right}, W_{middle}
- Only input: 5 $< o, l >$ tuples
- Punctuation matters: performance decreases when punctuation is removed
- Recall b/w .78 and .87 ($\tau_{sup} > 5$); precision .90 ($\tau_{sup} >> 4$)
- High precision possible (.96 with τ_t = .8); remaining problems come from NE recognition
- Pattern evaluation step responsible for most improvement over DIPRE

Summary: IE and template matching, learning

- Possible to learn simple relations from positive examples (Snowball)
- Possible to learn more diverse relations from annotated training corpus (Riloff)
- Even modest performance can be useful
  - Later manual verification
  - In circumstances where there would be no time to review source documents, so incomplete extracted information is better than none

Summary: IE Performance

Current methods perform well if

- Information to be extracted is expressed directly (no complex inference is required)
- Information is predominantly expressed in a relatively small number of forms
- Information is expressed locally within the text

Difference between IE and QA (next time):

- IE is domain dependent, open-domain QA is not

- Ellen Riloff, Automatically constructing a dictionary for information extraction tasks. In Proc. 11th Ann. Conference of Artificial Intelligence, p 811-816, 1993
- Eugene Agichtein, Luis Gravano: Snowball: Extracting Relations from Large Plain-Text Collections, Proceedings of the Fifth ACM International Conference on Digital Libraries, 2000