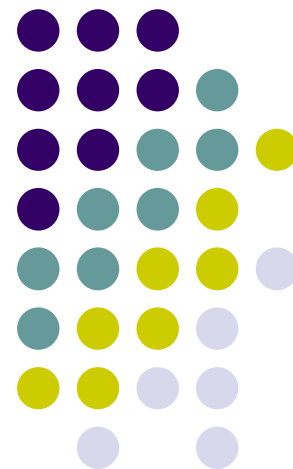
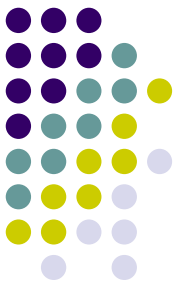


Learning Domain-Specific Information Extraction Patterns from the Web

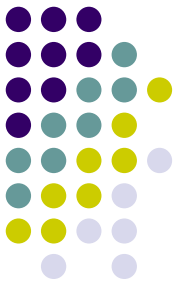
Siddharth Patwardhan and Ellen Riloff
University of Utah





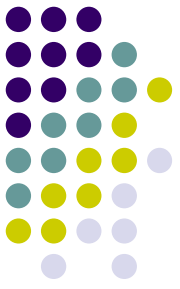
Motivation

- Many IE systems rely on extraction patterns or rules learned from domain-specific annotated training data.
- Domain-specific annotated corpora are expensive to create and are relatively small.
- A small training set limits the patterns that can be learned.



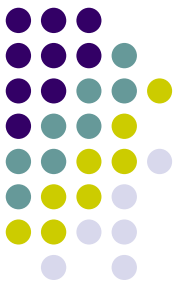
Goals of the Research

- Use the web to learn new extraction patterns not seen in training texts.
- Increase coverage of an IE system, using newly learned patterns.



Overview

- The Extraction Patterns
- The Seed System: AutoSlog-TS
- Data Collection
- The Learning Algorithm:
 - Learning Candidate Patterns
 - Semantic Affinity Filter
- Experiments and Results
- Conclusions

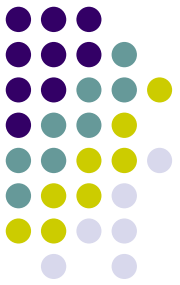


The Extraction Patterns

- Lexico-syntactic patterns based on an underlying shallow parse of the text.
- Patterns extract syntactic constituents.
- Optional selectional restrictions (semantic constraints) may be applied to the extractions.

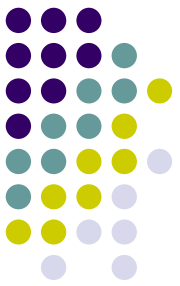
<subj> [HUMAN] ActVP(murdered)

The pirates brutally murdered their leader.



Extraction Pattern Types

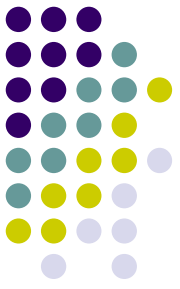
<subj> PassVP	<victim> <i>was brutally murdered</i>
<subj> ActVP	<perp> <i>murdered</i>
<subj> ActVP Dobj	<weapon> <i>caused massive damage</i>
<subj> ActInfVP	<perp> <i>tried to kill</i>
<subj> PassInfVP	<weapon> <i>was intended to kill</i>
<subj> AuxVP Dobj	<victim> <i>was a casualty</i>
<subj> AuxVP Adj	<victim> <i>is dead</i>



Extraction Pattern Types

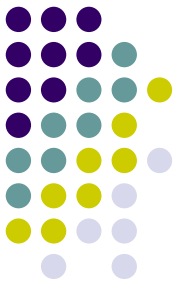
ActVP <dobj>	<i>bombed <target></i>
InfVP <dobj>	<i>to kill <victim></i>
ActInfVP <dobj>	<i>planned to bomb <target></i>
PassInfVP <dobj>	<i>was planned to kill <victim></i>
Subj AuxVP <dobj>	<i>fatality is <victim></i>
NP Prep <np>	<i>attack against <target></i>
ActVP Prep <np>	<i>killed quickly with <weapon></i>
PassVP Prep <np>	<i>was killed with <weapon></i>
InfVP Prep <np>	<i>to destroy with <weapon></i>
<possessive> NP	<i><victim>'s murder</i>

AutoSlog-TS: Seed System

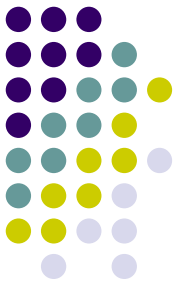


- Weakly supervised learner which learns extraction patterns using pre-classified texts.
- Requires two sets of texts:
 - Relevant to the domain.
 - Irrelevant texts.
- Manual review of learned patterns is required.

AutoSlog-TS

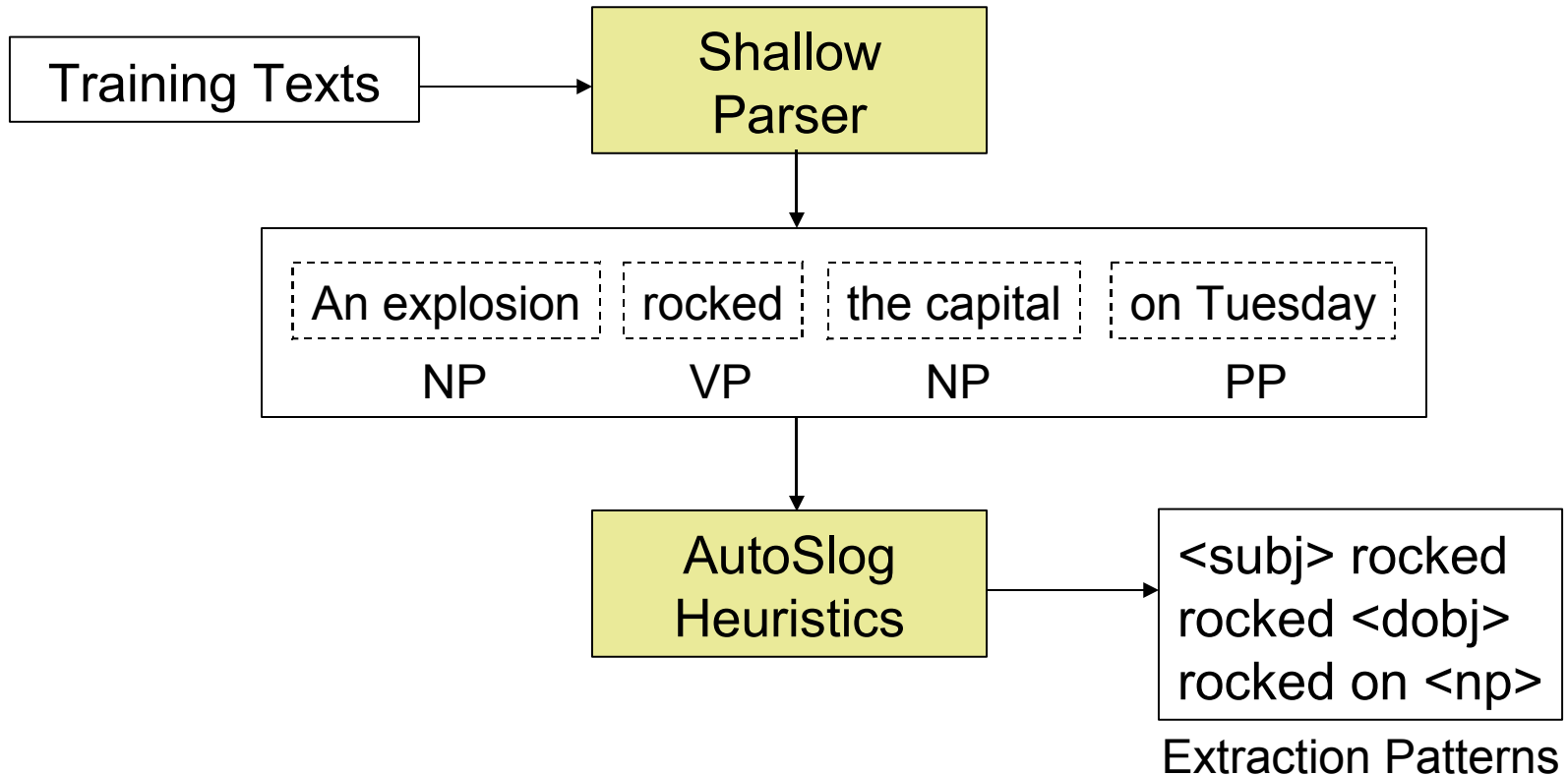


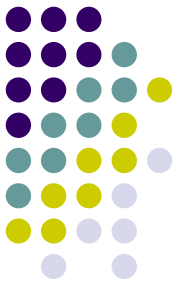
- First phase: generates every possible extraction pattern from all the texts.
- Second phase: gathers statistics for the extraction patterns, using the relevant and irrelevant texts.
- Ranked list is manually reviewed:
 - to keep the good patterns.
 - assign thematic roles to them.



AutoSlog-TS Overview

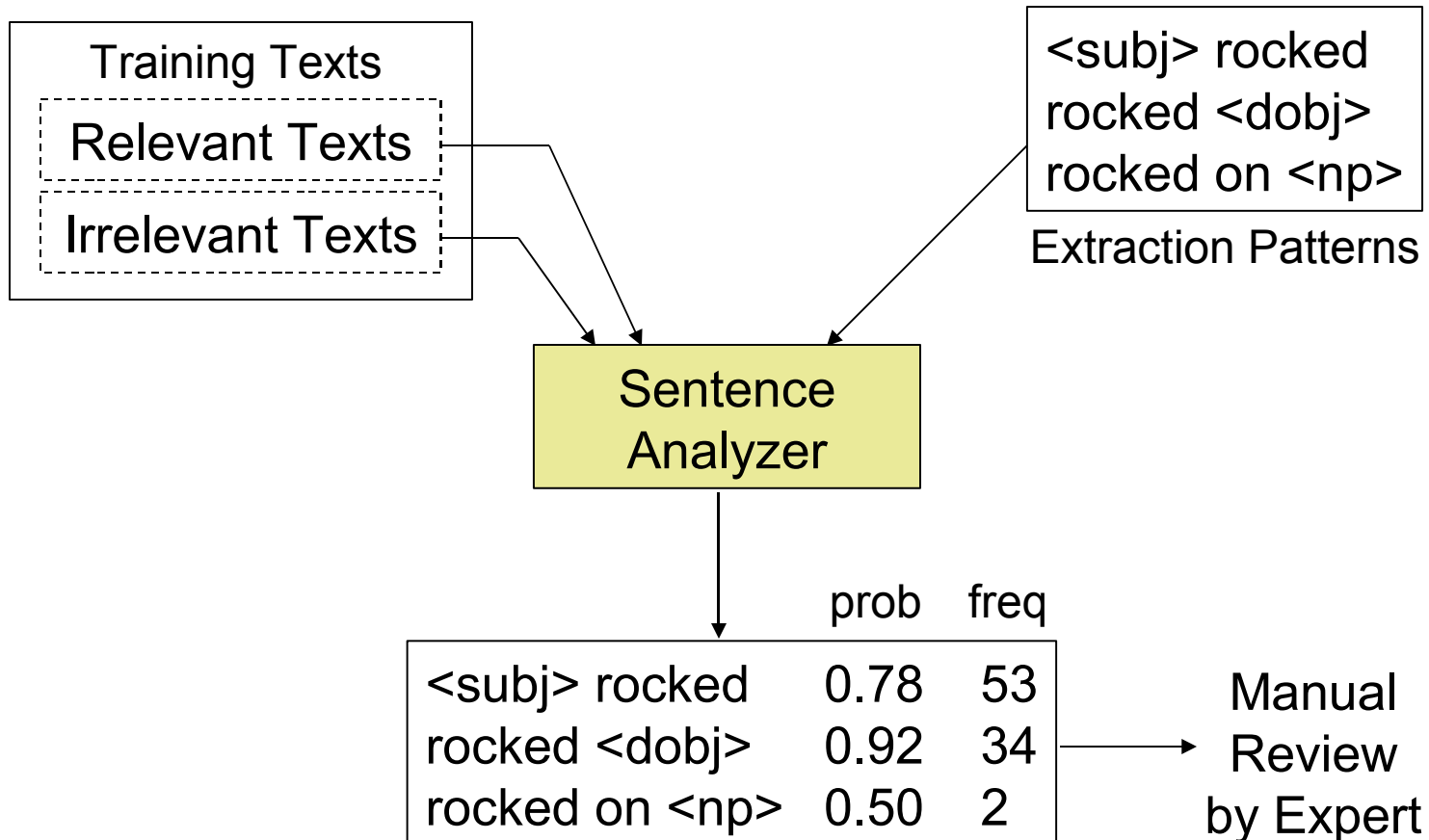
- Step 1:

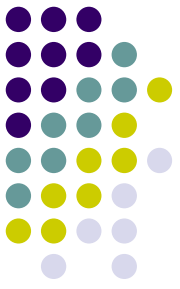




AutoSlog-TS Overview

- Step 2:



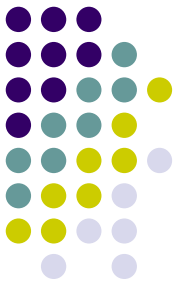


The RLogF Score

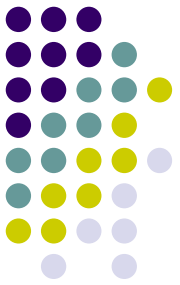
$$\text{RLogF} (pat) = \frac{freq_{rel} (pat)}{freq_{all} (pat)} \log_2 (freq_{rel} (pat))$$

- $freq_{rel}$ is the frequency of the extractions of pattern in relevant texts.
- $freq_{all}$ is the frequency of the extractions of pattern in all the texts.

AutoSlog-TS on MUC-4



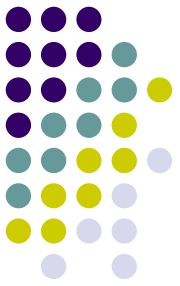
- We focused on the MUC-4 IE task – to extract information about Latin-American terrorist events.
- Using 1500 MUC-4 training documents, AutoSlog-TS we obtained 396 relevant terrorism patterns.
- Of these, 291 were *human victim* and *physical target* patterns.



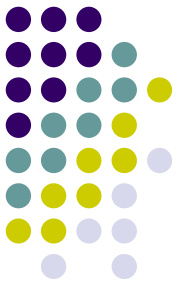
Checkpoint

- The Extraction Patterns
- The Seed System: AutoSlog-TS
- **Data Collection**
- The Learning Algorithm:
 - Learning Candidate Patterns
 - Semantic Affinity Filter
- Experiments and Results
- Conclusions

Data Collection



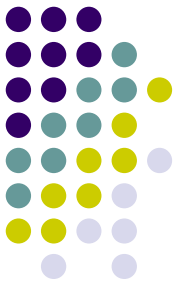
- Goal of this research is to acquire additional domain-specific patterns from the Web.
- We created a domain-specific corpus of terrorism-related CNN news articles.
- Used the *Google* search engine with hand-crafted queries to build the corpus.



Data Collection

- Each query consisted of:
 - name of a terrorist organization.
 - phrase related to a terrorist action.

Organization	Action
Al Qaeda, ELN, FARC, HAMAS, IRA	assassinated, assassination, blew up, bombed, bombing, bombs, explosion, hijacked, hijacking, injured, kidnapped, kidnapping, killed, murder, suicide bomber, wounded

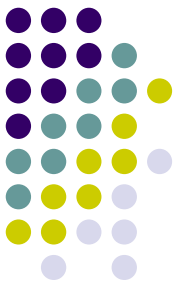


Data Collection

- Sample queries:

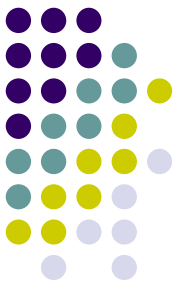
```
site:cnn.com "Al Qaeda" AND "assassination"  
site:cnn.com "HAMAS" AND "blew up"  
site:cnn.com "FARC" AND "kidnapped"
```

- Restricted the search to English language web pages.
- Removed transcripts of CNN TV shows.
- Collected 6,182 CNN news articles.



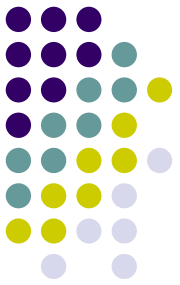
Cleaning the Texts

- Used tool called *HTMLParser* to create HTML parse tree of the web pages.
- Relied on the uniformity of CNN web pages to get the appropriate text.
- Deleted extremely large and extremely small documents.
- Obtained a text corpus of 5,618 cleaned documents.

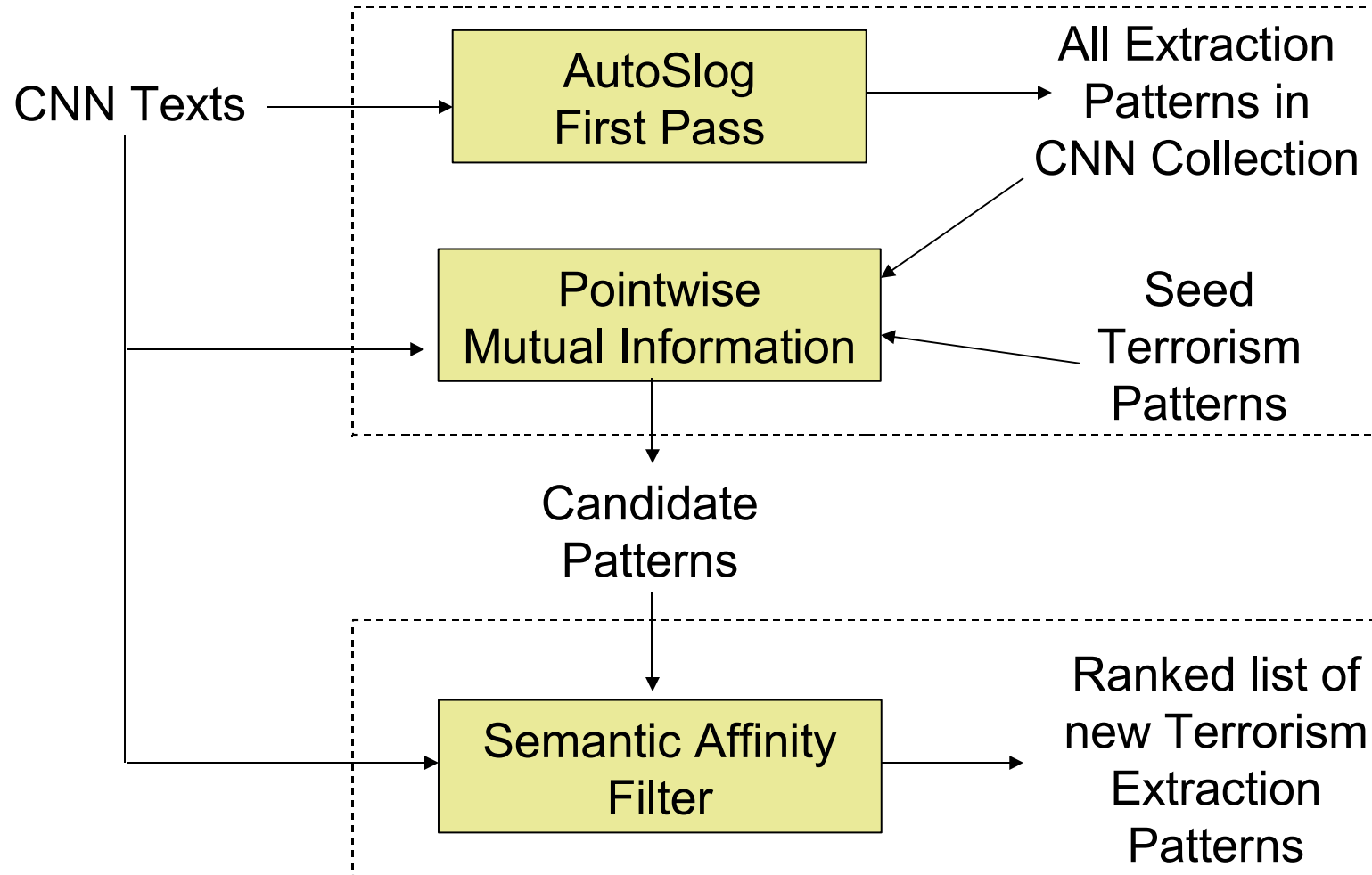


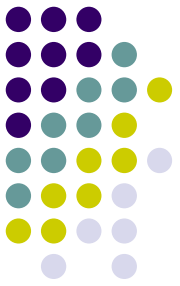
The Learning Algorithm

- Basic approach is to identify relevant sentences within the CNN texts using seed patterns.
- Then find new patterns that occur in these sentences more often than chance.
- The hypothesis is that patterns that occur more often in the relevant regions are likely to be of interest.

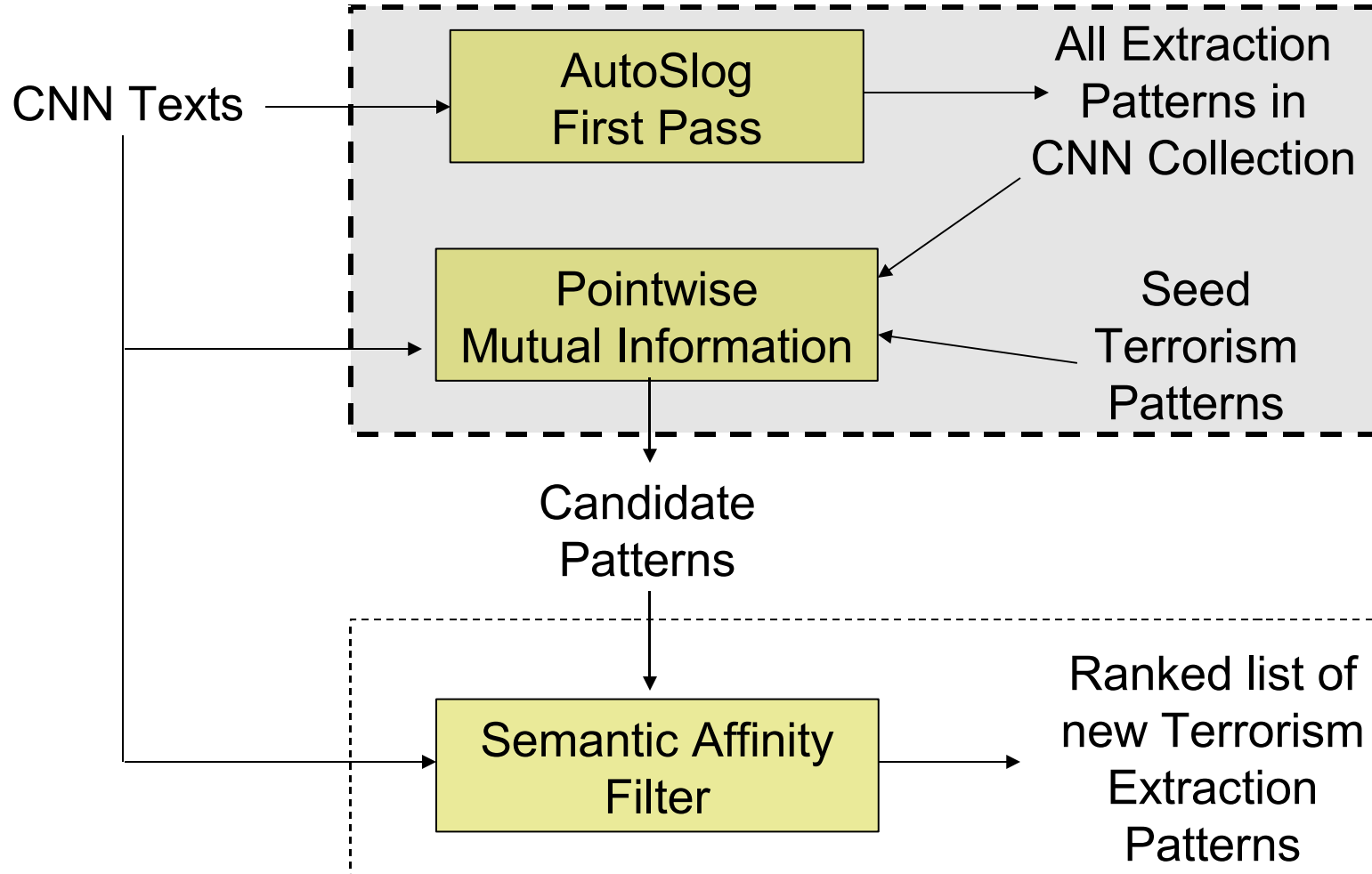


The Learning Algorithm

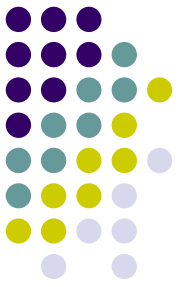




Identifying Candidate Patterns

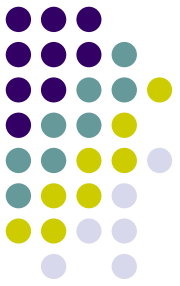


Identifying Candidate Patterns



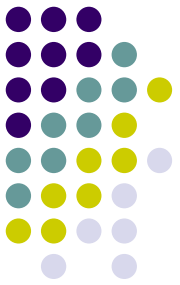
- AutoSlog-TS: generates every possible pattern from the CNN texts.
- 147,712 patterns were created.
- Computed statistical correlation (pointwise mutual information) of each pattern with a set of *seed patterns*.

Pointwise Mutual Information



$$\text{PMI}(pat, S) = \log \left(\frac{P(pat, S)}{P(pat) \cdot P(S)} \right)$$

- $P(pat, S)$ is the probability of candidate pattern pat and a seed pattern S occurring in the same sentence.
- $P(pat)$ is the probability of pattern pat occurring in the corpus of text.



Candidate Patterns

<subj> killed sgt

<subj> burned flag

sympathizers of <np>

<subj> kills bystanders

rescued within <np>

<subj> threatened me

emperor since <np>

<subj> shot american

<subj> destroyed factories

explode after <np>

<subj> killed heir

<subj> shattered roof

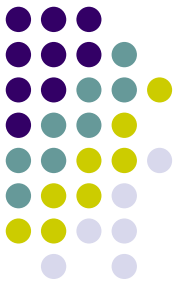
fled behind <np>

<subj> kills students

<subj> hit lodge

going to capture <dobj>

Candidate Patterns



<subj> killed sgt

<subj> burned flag

sympathizers of <np>

<subj> kills bystanders

rescued within <np>

<subj> threatened me

emperor since <np>

<subj> shot american

<subj> destroyed factories

explode after <np>

<subj> killed heir

<subj> shattered roof

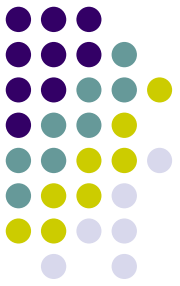
fled behind <np>

<subj> kills students

<subj> hit lodge

going to capture <dobj>

Candidate Patterns



<subj> killed sgt

<subj> burned flag

sympathizers of <np>

<subj> kills bystanders

rescued within <np>

<subj> threatened me

emperor since <np>

<subj> shot american

<subj> destroyed factories

explode after <np>

<subj> killed heir

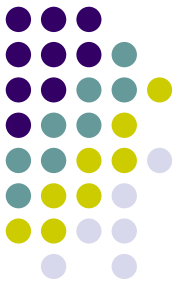
<subj> shattered roof

fled behind <np>

<subj> kills students

<subj> hit lodge

going to capture <dobj>



Candidate Patterns

<subj> killed sgt

<subj> burned flag

sympathizers of <np>

<subj> kills bystanders

rescued within <np>

<subj> threatened me

emperor since <np>

<subj> shot american

<subj> destroyed factories

explode after <np>

<subj> killed heir

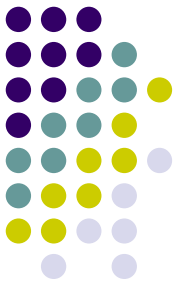
<subj> shattered roof

fled behind <np>

<subj> kills students

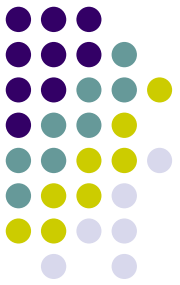
<subj> hit lodge

going to capture <dobj>

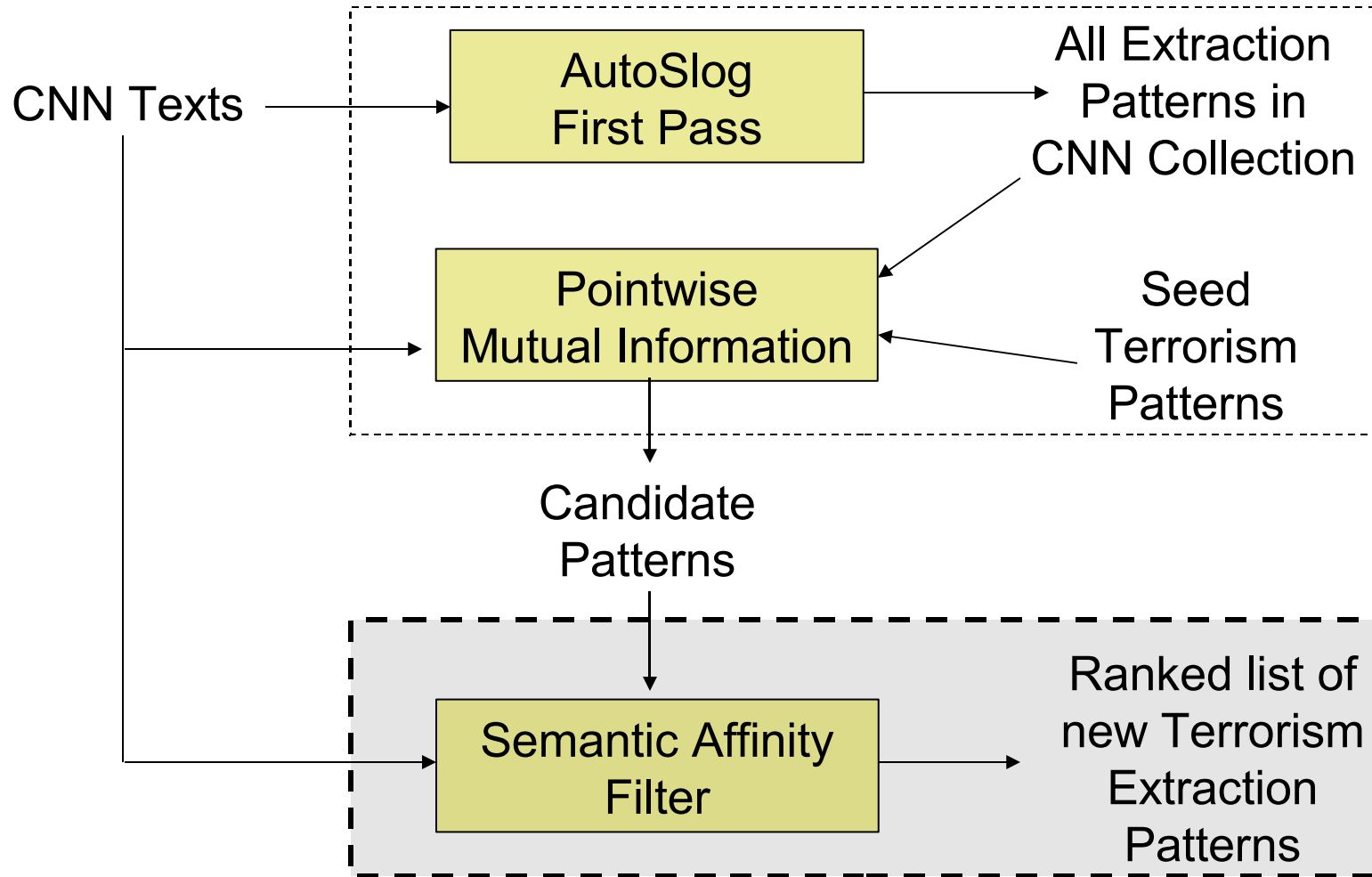


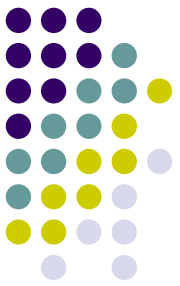
Candidate Patterns

- Some candidate patterns are related to terrorism, but do not extract any relevant information.
- The extractions from the candidate patterns are not assigned a thematic role (or an extraction slot).



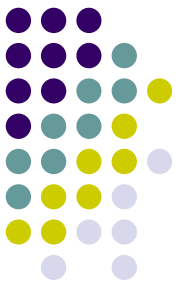
Semantic Affinity Filter





Semantic Affinity Filter

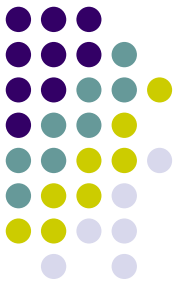
- We define a measure called Semantic Affinity – the tendency of a pattern to extract NPs of a semantic category.
- It serves two purposes:
 - Filters out candidate patterns with no semantic affinity to any category of interest.
 - Allows us to define a possible mapping between patterns and thematic roles (slot types).



Computing Semantic Affinity

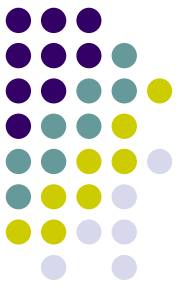
- Used the Sundance parser to identify the semantic class of extracted NPs.
- Counted the frequencies of semantic categories extracted by each candidate pattern.
- Semantic Affinity of a pattern p to a category c_i :

$$\text{affinity}(p, c_i) = \frac{\text{freq}(p, c_i)}{\sum_{j=1}^{|cat|} \text{freq}(p, c_j)} \log_2(\text{freq}(p, c_i))$$



Semantic Affinity

- Computed the semantic affinity of each candidate pattern with respect to the following categories:
 - *Human Victim*
 - *Physical Target*
 - *Weapon*
 - *Perpetrator Individual*
 - *Perpetrator Organization*
 - *Other*



Semantic Affinity Filter

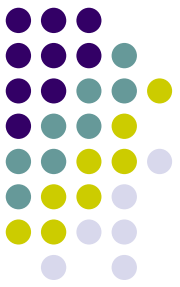
- Filtered candidate patterns not strongly associated with any relevant category.
- Deleted patterns whose semantic affinity for the *other* category was greater than the rest of the categories.

Target

missiles at <np>
<subj> fired missiles
bombs near <np>
fired into <np>
died on <np>

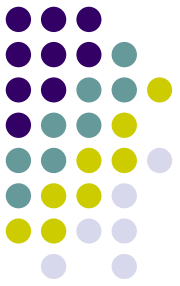
Victim

wounded in <np>
<subj> was identified
wounding <dobj>
<subj> wounding
identified <dobj>



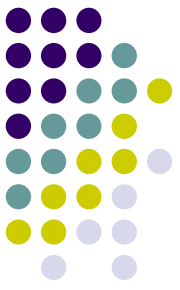
Checkpoint

- The Extraction Patterns
- The Seed System: AutoSlog-TS
- Data Collection
- The Learning Algorithm:
 - Learning Candidate Patterns
 - Semantic Affinity Filter
- **Experiments and Results**
- Conclusions



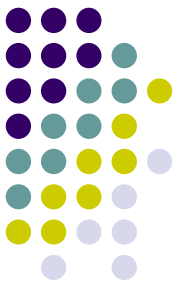
The MUC-4 IE Task

- To extract information about Latin-American terrorist events.
- 1,700 stories, with answer-key templates.
- We focused on two *string* slots:
 - Physical Targets
 - Human Victims



Experimental Setup

- MUC-4 Data Set:
 - 1,300 development documents (DEV).
 - Four test sets of 100 documents each (TST1, TST2, TST3, TST4).
- We used 1,500 texts (DEV + TST1 + TST2) for training, and 200 texts (TST3 + TST4) for testing.

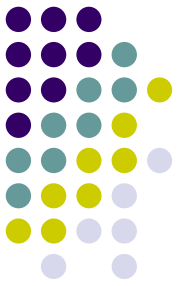


Baseline Results

- AutoSlog-TS system learned 291 target and victim patterns.
- Scored on the test set using *extraction level* scoring (no template generation), with *head noun* matching.

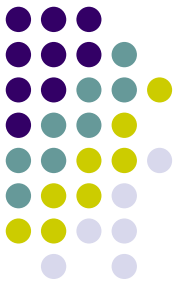
	Precision	Recall	F-measure
Target	0.425	0.642	0.511
Victim	0.498	0.517	0.507

Evaluating the New Patterns

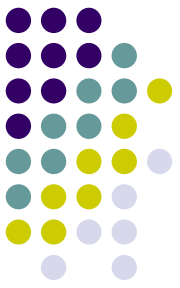


- We used all 396 terrorism patterns learned by AutoSlog-TS as *seed patterns* for our learning algorithm.
- Produced a list of new patterns ranked by PMI (correlation with seeds).
- Used a semantic affinity cutoff of 3.0.

Results for Target Patterns

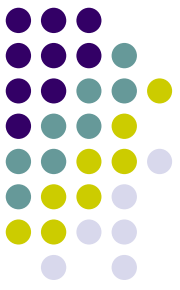


# of patterns	Precision	Recall	F-measure
baseline	0.425	0.642	0.511
50+baseline	0.420	0.642	0.508
100+baseline	0.419	0.650	0.510
150+baseline	0.415	0.650	0.507
200+baseline	0.412	0.667	0.509
250+baseline	0.401	0.691	0.507
300+baseline	0.394	0.691	0.502



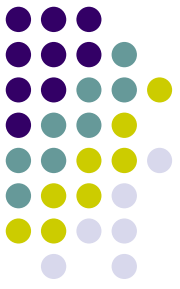
Results for Victim Patterns

# of patterns	Precision	Recall	F-measure
baseline	0.498	0.517	0.507
50+baseline	0.498	0.517	0.507
100+baseline	0.496	0.521	0.508
150+baseline	0.480	0.521	0.500
200+baseline	0.478	0.521	0.499
250+baseline	0.478	0.521	0.499
300+baseline	0.471	0.542	0.504



Conclusions

- It is possible to learn new extraction patterns from the Web, and use them to improve the coverage on a domain-specific IE task.
- Our approach produced a 5% increase in recall for targets, and a 3% increase in recall for victims – both at some loss in precision.



Future Work

- Plan to develop improved ranking methods for candidate patterns.
- Better semantic affinity measures.
- Possibly embed this approach in a bootstrapping mechanism.
- Try to automate the document collection process.

Questions?

