Generatin Coherent Event Schemas at Scale

Niranjan Balasubramanian, Stephen Soderland, Mausam, Oren Etzioni

University of Washington

Presented By: Jumana
1 Introduction

2 System Overview
   - Modeling Relational Co-occurrence
     - Relation Extraction
     - Rel-gram Database
     - Rel-grams Language Model
   - Schema Generation
     - Rel-graph Construction
     - Finding Related Tuples
     - Creating Actors and Relations

3 Evaluation
   - Related Work
   - Evaluation
   - Error Analysis
   - Future Work
Event Schemas

- A set of relations and actors that serve as arguments.
- Applications:
  - Extracting information related to events
  - Coreference
  - Summarization
  - Inference about temporal ordering and causality.
- Used to be manually created, and focused on specific domains of interest.
- Goal: automatically generate coherent event schemas on an open domain.
Event Schemas

- A set of relations and actors that serve as arguments.

- Applications:
  - Extracting information related to events
  - Coreference
  - Summarization
  - Inference about temporal ordering and causality.

- Used to be manually created, and focused on specific domains of interest.

- Goal: automatically generate coherent event schemas on an open domain.
Event Schemas

- A set of relations and actors that serve as arguments.
- Applications:
  - Extracting information related to events
  - Coreference
  - Summarization
  - Inference about temporal ordering and causality.
- Used to be manually created, and focused on specific domains of interest.
- Goal: automatically generate coherent event schemas on an open domain.
Event Schemas

- A set of relations and actors that serve as arguments.

- Applications:
  - Extracting information related to events
  - Coreference
  - Summarization
  - Inference about temporal ordering and causality.

- Used to be manually created, and focused on specific domains of interest.

- **Goal**: automatically generate coherent event schemas on an open domain.
Frequently co-occurring relations in text capture relatedness.
Extracted relational triples using OLLIE Open IE (Mausam et al., 2012) from a set of 1.8 Million New York Times articles.

- 320K tuples with frequency $\geq 3$.
- The triples are of the form: $(\text{Arg}_1, \text{Relation}, \text{Arg}_2)$

Example:

Sentence: He cited a new study that was released by UCLA in 2008.

Produced Tuples:
Extracted relational triples using OLLIE Open IE (Mausam et al., 2012) from a set of 1.8 Million New York Times articles.

- 320K tuples with frequency $\geq 3$.
- The triples are of the form: $(Arg1, Relation, Arg2)$
- Example:
  - Sentence: He cited a new study that was released by UCLA in 2008.
  - Produced Tuples:
    - (He, cited, a new study)
    - (a new study, was released by, UCLA)
    - (a new study, was released in, 2008)
Extracted relational triples using **OLLIE Open IE** (Mausam et al., 2012) from a set of 1.8 Million New York Times articles.

- 320K tuples with frequency \( \geq 3 \).
- The triples are of the form: \((Arg1, Relation, Arg2)\)
- Example:
  - Sentence: He cited a new study that was released by UCLA in 2008.
  - Produced Tuples:
    - (He, cited, a new study)
    - (a new study, was released by, UCLA)
    - (a new study, was released in, 2008)
A relation phrase is represented by its stemmed head verb plus any prepositions.

Tuple arguments are also represented as stemmed head nouns.

Record semantic types of the arguments (to avoid sparsity):

- Selected 29 semantic types from WordNet: person, organization, location, time_unit, number, amount, group, business, executive, leader, effect, activity, game, sport, device, equipment, structure, building, substance, nutrient, drug, illness, organ, animal, bird, fish, art, book, and publication.
- Stanford Named Entity Recognizer (Finkel et al., 2005) was used to assign types to arguments.
- In addition, they looked up the argument in WordNet and used the first three senses, if they map to the target semantic types.
Extracted Tuples

- A relation phrase is represented by its stemmed head verb plus any prepositions.
- Tuple arguments are also represented as stemmed head nouns.
- Record semantic types of the arguments (to avoid sparsity):
  - Selected 29 semantic types from WordNet: person, organization, location, time_unit, number, amount, group, business, executive, leader, effect, activity, game, sport, device, equipment, structure, building, substance, nutrient, drug, illness, organ, animal, bird, fish, art, book, and publication.
  - Stanford Named Entity Recognizer (Finkel et al., 2005) was used to assign types to arguments.
  - In addition, they looked up the argument in WordNet and used the first three senses, if they map to the target semantic types.
Extracted Tuples

- A relation phrase is represented by its stemmed head verb plus any prepositions.
- Tuple arguments are also represented as stemmed head nouns.
- Record **semantic types** of the arguments (to avoid sparsity):
  - Selected 29 semantic types from *WordNet*: person, organization, location, time_unit, number, amount, group, business, executive, leader, effect, activity, game, sport, device, equipment, structure, building, substance, nutrient, drug, illness, organ, animal, bird, fish, art, book, and publication.
  - *Stanford Named Entity Recognizer* (Finkel et al., 2005) was used to assign types to arguments.
  - In addition, they looked up the argument in *WordNet* and used the first three senses, if they map to the target semantic types.
A relation phrase is represented by its stemmed head verb plus any prepositions.

Tuple arguments are also represented as stemmed head nouns.

Record **semantic types** of the arguments (to avoid sparsity):

- Selected **29** semantic types from **WordNet**: person, organization, location, time_unit, number, amount, group, business, executive, leader, effect, activity, game, sport, device, equipment, structure, building, substance, nutrient, drug, illness, organ, animal, bird, fish, art, book, and publication.

- **Stanford Named Entity Recognizer** (Finkel et al., 2005) was used to assign types to arguments.

- In addition, they looked up the argument in **WordNet** and used the first three senses, if they map to the target semantic types.
A relation phrase is represented by its stemmed head verb plus any prepositions. Tuple arguments are also represented as stemmed head nouns. Record **semantic types** of the arguments (to avoid sparsity):

- Selected 29 semantic types from **WordNet**: person, organization, location, time_unit, number, amount, group, business, executive, leader, effect, activity, game, sport, device, equipment, structure, building, substance, nutrient, drug, illness, organ, animal, bird, fish, art, book, and publication.

**Stanford Named Entity Recognizer** (Finkel et al., 2005) was used to assign types to arguments. In addition, they looked up the argument in **WordNet** and used the first three senses, if they map to the target semantic types.
(He, cite, study)
(He, cite, <activity>)
(<person>, cite, study)
(<person>, cite, <activity>)
(study, be release by, UCLA)
(study, be release by, <organization>)
(study, be release by, 2008)
(study, be release by, <time_unit>)
(<activity>, be release by, UCLA)
...
Rel-gram Database

- (SQL) database to hold co-occurrence statistics of pairs of tuples found in each document.

<table>
<thead>
<tr>
<th>Tuples Table</th>
<th>BigramCounts Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Arg1</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>13</td>
<td>bomb</td>
</tr>
<tr>
<td>14</td>
<td>bomb</td>
</tr>
<tr>
<td>15</td>
<td>bomb</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>87</td>
<td>bomb</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>92</td>
<td>&lt;loc&gt;</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Rel-gram Database - Tables

- **Tuples**(320K): tuples and their frequencies.
- **Bi-gramCounts**(1.1M): directional co-occurrence frequency at distance $k$, and the number of times the same argument was present in a pair of tuples.
- An argument pair is equal if:
  - They are from the same token sequence in the source sentence.
  - One argument is a co-reference mention of the other (Stanford Co-reference system).
Tuples(320K): tuples and their frequencies.

Bi-gramCounts(1.1M): directional co-occurrence frequency at distance $k$, and the number of times the same argument was present in a pair of tuples.

An argument pair is equal if:
  - They are from the same token sequence in the source sentence.
  - One argument is a co-reference mention of the other (Stanford Co-reference system).
Tuples($320K$): tuples and their frequencies.

Bi-gramCounts($1.1M$): directional co-occurrence frequency at distance $k$, and the number of times the same argument was present in a pair of tuples.

An argument pair is equal if:

- They are from the same token sequence in the source sentence.
- One argument is a co-reference mention of the other (Stanford Co-reference system).
Estimate bi-gram conditional probabilities of tuples:

\[ P_k(T' | T) = \frac{#(T, T', k) + \delta}{\sum_{T'' \in V} #(T, T'', k) + \delta \cdot |V|} \]  

\[ P(T' | T) = \frac{\sum_{k=1}^{10} \alpha^k P_k(T' | T)}{\sum_{k=1}^{10} \alpha^k} \]

\( \delta \) was used to discount estimates from low-frequency tuples.
\( \alpha \) was used to give more weight to closer tuples (smaller distance).
Use Rel-grams to identify relations and actors of particular event.

Steps:

- Rel-graph construction.
- Finding related tuples.
- Creating actors and relations.
Schema Generation

Use Rel-grams to identify relations and actors of particular event.

Steps:
- Rel-graph construction.
- Finding related tuples.
- Creating actors and relations.
Rel-graph Construction

- Undirected weighted graph $G = (V, E)$. $V$ are relation tuples.
  An edge between $T$ and $T'$ is weighted by the symmetric conditional probability:

$$SCP(T, T') = P(T|T') \times P(T'|T) \quad (3)$$
Use high connectivity nodes as seeds:

- sort nodes by the sum of their top 25 edge weights,
- take the top portion of this list.

For each seed \((Q)\), build a sub-graph \((G_Q)\) containing \(Q\)’s neighbors (within 2 steps from \(Q\)).
Finding Related Tuples

- Use high connectivity nodes as seeds:
  - sort nodes by the sum of their top 25 edge weights,
  - take the top portion of this list.

- For each seed \((Q)\), build a sub-graph \((G_Q)\) containing \(Q\)'s neighbors (within 2 steps from \(Q\)).
Finding Related Tuples

Detecting Strongly Connected Nodes within $G_Q$:

- Use **Personalized PageRank** algorithm (Haveliwala, 2002).
- Returns ranks of various nodes with respect to a given query node.
- An iterative process:
  - initialization: page rank of $Q = 1$ and zero to all the rest;
  - compute the page rank according to the given equations until they converge to within a small epsilon.
Detecting Strongly Connected Nodes within $G_Q$:

- Use **Personalized PageRank** algorithm (Haveliwala, 2002).
- Returns ranks of various nodes with respect to a given query node.
- An iterative process:
  - initialization: page rank of $Q = 1$ and zero to all the rest;
  - compute the page rank according to the given equations until they converge to within a small epsilon.
Creating Actors and Relations

- Take the top \( n \) tuples from \( G_Q \), according to their page rank.
- For each tuple \( T: (\text{Arg}_1, \text{Rel}, \text{Arg}_2) \) in \( G_Q \), record 2 actors: \( A_1, A_2 \) and add \( \text{Rel} \) to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constrains.
  - Merge \( A_1 \) and \( A_2 \) if they are connected to the same actor \( A_3 \) through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g., location-date).
  - Do not merge actors if it results in a relation where the same actor is both Arg_1 and Arg_2.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within \( G_Q \).
Creating Actors and Relations

- Take the top $n$ tuples from $G_Q$, according to their page rank.
- For each tuple $T: (Arg1, Rel, Arg2)$ in $G_Q$, record 2 actors: $A_1, A_2$ and add $Rel$ to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constraints.
  - Merge $A_1$ and $A_2$, if they are connected to the same actor $A_3$ through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g., location-date).
  - Do not merge actors if it results in a relation where the same actor is both $Arg1$ and $Arg2$.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within $G_Q$. 

N. Balasubramanian, S. Soderland, Mausam, O. Etzioni
Generating Coherent Event Schemas at Scale
Creating Actors and Relations

- Take the top $n$ tuples from $G_Q$, according to their page rank.
- For each tuple $T: (Arg_1, Rel, Arg_2)$ in $G_Q$, record 2 actors: $A_1, A_2$ and add $Rel$ to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constrains.
  - Merge $A_1$ and $A_2$, if they are connected to the same actor $A_3$ through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g. location-date).
  - Do not merge actors if it results in a relation where the same actor is both $Arg_1$ and $Arg_2$.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within $G_Q$. 

N. Balasubramanian, S. Soderland, Mausam, O. Etzioni

Generating Coherent Event Schemas at Scale
Creating Actors and Relations

- Take the top $n$ tuples from $G_Q$, according to their page rank.
- For each tuple $T: (Arg1, Rel, Arg2)$ in $G_Q$, record 2 actors: $A_1, A_2$ and add $Rel$ to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constrains.
  - Merge $A_1$ and $A_2$, if they are connected to the same actor $A_3$ through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g. location-date).
  - Do not merge actors if it results in a relation where the same actor is both $Arg1$ and $Arg2$.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within $G_Q$.  

N. Balasubramanian, S. Soderland, Mausam, O. Etzioni  
Generating Coherent Event Schemas at Scale
Creating Actors and Relations

- Take the top $n$ tuples from $G_Q$, according to their page rank.
- For each tuple $T: (Arg_1, Rel, Arg_2)$ in $G_Q$, record 2 actors: $A_1, A_2$ and add $Rel$ to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constraints.
  - Merge $A_1$ and $A_2$, if they are connected to the same actor $A_3$ through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g. location-date).
  - Do not merge actors if it results in a relation where the same actor is both $Arg_1$ and $Arg_2$.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within $G_Q$. 

N. Balasubramanian, S. Soderland, Mausam, O. Etzioni
Generating Coherent Event Schemas at Scale
Creating Actors and Relations

- Take the top $n$ tuples from $G_Q$, according to their page rank.
- For each tuple $T: (Arg1, Rel, Arg2)$ in $G_Q$, record 2 actors: $A_1$, $A_2$ and add $Rel$ to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constrains.
  - Merge $A_1$ and $A_2$, if they are connected to the same actor $A_3$ through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g. location-date).
  - Do not merge actors if it results in a relation where the same actor is both $Arg1$ and $Arg2$.
- Sort tuples by the average page rank of the original tuples, reflecting their importance within $G_Q$. 

N. Balasubramanian, S. Soderland, Mausam, O. Etzioni
Generatin Coherent Event Schemas at Scale
Creating Actors and Relations

- Take the top \( n \) tuples from \( G_Q \), according to their page rank.
- For each tuple \( T: (Arg1, Rel, Arg2) \) in \( G_Q \), record 2 actors: \( A_1, A_2 \) and add \( Rel \) to the list of relations they participate in.
- Merge actors:
  - Merge actors that have non-zero equality constrains.
  - Merge \( A_1 \) and \( A_2 \), if they are connected to the same actor \( A_3 \) through the same relation. Example: (lawsuit, file by, company) and (suit, file by, company).
  - There are semantic type pairs that cannot be merged (e.g. location-date).
  - Do not merge actors if it results in a relation where the same actor is both \( Arg1 \) and \( Arg2 \).
- Sort tuples by the average page rank of the original tuples, reflecting their importance within \( G_Q \).
Related Work - Cheung et al. (2013)

- Probabilistic model.
- Requires performing joint probability estimation using EM,
- hence, limit scaling to large corpora.
Their system:
- fully automatic,
- domain-independent,
- scales to large text corpora.

Pair-wise representation:
- subject-verb pairs are treated independently from verb-object pairs.

Problematic: non meaningful subject-verb-object triples.
Weakness:

- Some schemas lack a common topic.
  (e.g. spreading fire and spreading disease are in the same event schema)
- Distinct roles are incorrectly mixed into a single actor.
  (non-valid tuple: child plant bomber)
- Chambers and Jurafsky (2011): learn domain specific event templates and associated extractors.
Weakness:

- Some schemas lack a common topic.
  (e.g. spreading fire and spreading disease are in the same event schema)
- Distinct roles are incorrectly mixed into a single actor.
  (non-valid tuple: child plant bomber)

Chambers and Jurafsky (2011): learn domain specific event templates and associated extractors.
• Weakness:
  • Some schemas lack a common topic.
    (e.g. spreading fire and spreading disease are in the same event schema)
  • Distinct roles are incorrectly mixed into a single actor.
    (non-valid tuple: child plant bomber)

• Chambers and Jurafsky (2011): learn domain specific event templates and associated extractors.
Evaluation

Compared against Chambers’s event schemas:

- **Metrics:**
  - Schema pertains to a coherent topic.
  - Actors play a coherent role in that event.
  - Tuple validity.

- **Used Amazon Mechanical Turk workers:**
  - Topic coherence: 92% vs. 82% in favor of the suggested system.
  - Actor coherence: 81% vs. 59% in favor of the suggested system.
  - Tuple validity: 92% vs. 61% in favor of the suggested system.
Evaluation

Compared against Chambers’s event schemas:

- **Metrics:**
  - Schema pertains to a coherent topic.
  - Actors play a coherent role in that event.
  - Tuple validity.

- **Used Amazon Mechanical Turk workers:**
  - Topic coherence: 92% vs. 82% in favor of the suggested system.
  - Actor coherence: 81% vs. 59% in favor of the suggested system.
  - Tuple validity: 92% vs. 61% in favor of the suggested system.
Due to mismatched actors:
- Mismatch: 56% of the invalid tuples (5% of all tuples).
- A general type may have an instance that does not play the same role as other instances.
- Example: \((A_1, \text{graduated from}, A_2)\), \(A_2\) includes “church” \(\Rightarrow\) invalid tuple

Due to extraction errors:
- Errors: 47% of the invalid tuples.
- Some \(n\)-ary relations are misanalysed as binary.
- Example: “Mr. Diehl spends more time ... than the commissioner” (Mr. Diehl, spend than, commissioner).
Error Analysis

Due to mismatched actors:
- Mismatch: 56% of the invalid tuples (5% of all tuples).
- A general type may have an instance that does not play the same role as other instances.
- Example: \((A_1, \text{graduated from}, A_2)\), \(A_2\) includes “church” \(\Rightarrow\) invalid tuple

Due to extraction errors:
- Errors: 47% of the invalid tuples.
- Some \(n\)-ary relations are misanalysed as binary.
- Example: “Mr. Diehl spends more time . . . than the commissioner” (Mr. Diehl, spend than, commissioner).
Future Work:

- Use event schemas to create an open-domain event extractor.