Zero-Shot Transfer Learning for Event Extraction

Lifu Huang¹, Heng Ji¹, Kyunghyun Cho², Ido Dagan³, Sebastian Riedel⁴, Clare R. Voss⁵

¹ Rensselaer Polytechnic Institute
² New York University, ³ Bar-Ilan University,
⁴ University of College London,
⁵ Army Research Laboratory
Background

- Traditional Event Extraction
  - based on predefined event schema and rich features encoded from annotated event
  - *Pros*: extract high quality events for predefined types
  - *Cons*: require large amount of human annotations and cannot extract event mentions for new event types

**Traditional Event Extraction Pipeline**

*Consumer 1*: I want an event extractor for “Transport”
*Annotators*: We will annotate 500 documents
*System Developer*: I’ll train a classifier

... 

*Consumer 2*: I want an event extractor for “Attack”
*Annotators*: We will annotate 500 documents
*System Developer*: I’ll train a classifier

...
Background

- **Zero Shot Transfer Learning**
  - Learning a regression function between object (e.g., image, entity) semantic space and label semantic space based on annotated data for seen labels
  - The regression model can be used to predict the unseen labels for any given image

Andrea Frome, Greg S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc Aurelio Ranzato, Tomas Mikolov, *DeViSE: A Deep Visual-Semantic Embedding Model*
Motivation

- Zero Shot Learning for Event Extraction
  - both event mentions and types have rich semantics and structures, which can specify their consistency and connections

E1. The Government of China has ruled Tibet since 1951 after dispatching troops to the Himalayan region in 1950.

E2. Iranian state television stated that the conflict between the Iranian police and the drug smugglers took place near the town of mirjaveh.
Approach Overview
Approach Details

- **Trigger and Argument Identification**
  - Trigger Identification
    - AMR parsing and FrameNet verbs/nominal lexical units
  - Argument Identification
    - Subset of AMR relations

<table>
<thead>
<tr>
<th>Categories</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Roles</td>
<td>ARG0, ARG1, ARG2, ARG3, ARG4</td>
</tr>
<tr>
<td>None-Core Roles</td>
<td>mod, location, instrument, poss, manner, topic, medium, prep-X</td>
</tr>
<tr>
<td>Temporal</td>
<td>year, duration, decade, weekday, time</td>
</tr>
<tr>
<td>Spatial</td>
<td>destination, path, location</td>
</tr>
</tbody>
</table>

- **Event and Type Structure Construction**
Approach Details

- **Structure Composition and Representation**
  - **Event Mention Structure**
    - We use a matrix $M_\lambda$ to represent each AMR relation $\lambda$, and compose its semantics with two concepts for each tuple:
      
      \[ u = \langle w_1, \lambda, w_2 \rangle \quad \text{e.g.,} \quad \langle \text{dispatch-01, :ARG0, China} \rangle \]

      \[
      V_u = f ([V_{w_1}; V_{w_2}] \cdot M_\lambda)
      \]

  - **Event Type Structure**
    - Similarly, we assume an implicit relation exists between any pair of type and argument, and use a tensor $U^{[1:2d]}$ to represent it, and compose its semantics with each pair of type and argument role
      
      \[ u' = \langle y, r \rangle \quad \text{e.g.,} \quad \langle \text{Transport\_Person, Person} \rangle \]

      \[
      V_{u'} = f ([V_y; V_r]^T \cdot U^{[1:2d]} \cdot [V_y; V_r])
      \]
Approach Details

- **Joint Event Mention and Type Label Embedding**
  - Representation learning for each event mention structure and type structure
    - Take each structure (a sequence of tuples) as input, and encode each event mention and type structure into a vector representation using a weight-sharing *Convolutional Neural Network* (CNN)
  - Align the vector representations of each event mention structure with its corresponding event type structure
    - Minimize their distance within a share vector space
    - Over-fitting to seen types: seen types are usually very limited
Approach Details

- **Joint Event Mention and Type Label Embedding**
  - To avoid over-fitting for seen types
    - Add ‘negative’ event mentions into training
    - Negative event mentions: the mentions that are not annotated with any seen types, namely other. Extracted from the event mention clusters generated by Huang et. al. (2016)
  - Loss function

\[
L_1^d(t, y) = \begin{cases} 
\max_{j \in Y, j \neq y} \max\{0, m - C_{t,y} + C_{t,j}\}, & y \neq \text{Other} \\
\max_{j \in Y', j \neq y'} \max\{0, m - C_{t,y'} + C_{t,j}\}, & y = \text{Other}
\end{cases}
\]

\[
C_{t,y} = \cos([V_t; V_{S_t}],[V_y; V_{S_y}])
\]

where \(Y\) is the positive event type for the candidate trigger \(t\), \(Y\) is the type set of the event ontology, \(Y'\) is the seen type set. \(Y'\) is the type which ranks the highest among all event types for event mention \(t\)
Approach Details

- **Joint Event Argument and Role Embedding**
  - Mapping between argument and role path
    - Argument path: e.g., *dispatch01* -> :Arg0 -> *China*
    - Role path: *Transport_person* -> *Agent*
  - Learn path representations using two weight-sharing CNNs

- Loss function

\[
L_2^d(a, r) = \begin{cases} 
\max_{j \in R_y, j \neq r} \max \{0, m - C_{a,r} + C_{a,j}\} & r \neq \text{Other} \\
\max_{j \in R_{y'}, j \neq r'} \max \{0, m - C_{a,r'} + C_{a,j}\} & r | y = \text{Other} 
\end{cases}
\]

where \( r \) is the positive argument role for the candidate argument \( a \), \( R_y \) and \( R_{y'} \) are the set of argument roles which are predefined for trigger type \( y \) and all seen types \( y' \). \( r' \) is argument role which ranks the highest for \( a \) when \( a \) or \( y \) is annotated as \( \text{Other} \).
Evaluation

- **Zero-Shot Classification for ACE Events**
  - Given trigger and argument boundaries, use a subset of ACE types for training, and remained types for testing
  - Seen types for each experiment setting

<table>
<thead>
<tr>
<th>Setting</th>
<th>Top-N</th>
<th>Seen Types for Training/Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>Attack</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>Attack, Transport, Die</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>Attack, Transport, Die, Meet, Arrest-Jail</td>
</tr>
<tr>
<td>D</td>
<td>10</td>
<td>Attack, Transport, Die, Meet, Arrest-Jail, Transfer-Money, Sentence, Elect, Transfer-Ownership, End-Position</td>
</tr>
</tbody>
</table>
Evaluation

- Zero-Shot Classification for ACE Events
  - Statistics for Positive/Negative instances on Training, Development, and Test sets for each experiment setting
  - Negative instances are sampled from the trigger and argument clustering output of (Huang et. al., 2016)

<table>
<thead>
<tr>
<th>Setting Index</th>
<th>Training</th>
<th>Development</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># of Types/ Roles</td>
<td># of Events</td>
<td># of Arguments</td>
</tr>
<tr>
<td>A</td>
<td>1/5</td>
<td>953/900</td>
<td>894/1,097</td>
</tr>
<tr>
<td>B</td>
<td>3/14</td>
<td>1,803/1,500</td>
<td>2,035/1,791</td>
</tr>
<tr>
<td>C</td>
<td>5/18</td>
<td>2,033/1,300</td>
<td>2,281/1,503</td>
</tr>
<tr>
<td>D</td>
<td>10/37</td>
<td>2537/700</td>
<td>2,816/879</td>
</tr>
</tbody>
</table>
Evaluation

- **Zero-Shot Classification for ACE Events**
  - Hit@K performance on trigger and argument classification
  - Hit@K Accuracy: the correct label occurs within the top K ranked output labels
  - WSD-Embedding: directly map event triggers and arguments to event types and argument roles according to their cosine similarity of word sense embeddings

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit@k Trigger Typing (%)</th>
<th>Hit@k Argument Typing (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>WSD-Embedding</td>
<td>1.73</td>
<td>13.01</td>
</tr>
<tr>
<td>Transfer A</td>
<td>3.98</td>
<td>23.77</td>
</tr>
<tr>
<td>Transfer B</td>
<td>7.04</td>
<td>12.48</td>
</tr>
<tr>
<td>Transfer C</td>
<td>20.05</td>
<td>34.66</td>
</tr>
<tr>
<td>Transfer D</td>
<td>33.47</td>
<td>51.40</td>
</tr>
</tbody>
</table>
Evaluation

- Zero-Shot Classification for ACE Events
  - Training subtypes of Justice: *Arrest-Jail, Convict, Charge-Indict, Execute*
  - Performance on Various Unseen Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Subtype</th>
<th>Hit@k Trigger Typing%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Justice</td>
<td>Sentence</td>
<td>68.29</td>
</tr>
<tr>
<td>Justice</td>
<td>Appeal</td>
<td>67.50</td>
</tr>
<tr>
<td>Justice</td>
<td>Release-Parole</td>
<td>73.91</td>
</tr>
<tr>
<td>Conflict</td>
<td>Attack</td>
<td>26.47</td>
</tr>
<tr>
<td>Transaction</td>
<td>Transfer-Money</td>
<td>48.36</td>
</tr>
<tr>
<td>Business</td>
<td>Start-Org</td>
<td>0</td>
</tr>
<tr>
<td>Movement</td>
<td>Transport</td>
<td>2.60</td>
</tr>
<tr>
<td>Personell</td>
<td>End-Position</td>
<td>9.09</td>
</tr>
<tr>
<td>Life</td>
<td>Injure</td>
<td>87.64</td>
</tr>
<tr>
<td>Contact</td>
<td>Phone-Write</td>
<td>60.78</td>
</tr>
</tbody>
</table>
Evaluation

- Event Extraction for ACE Types
  - Target Event Ontology: ACE(33 types)+FrameNet (1161 frames)
  - Seen types for training: 10 ACE types
  - Performance on ACE types

<table>
<thead>
<tr>
<th>Setting</th>
<th>Method</th>
<th>Trigger Identification (%)</th>
<th>Trigger Typing (%)</th>
<th>Arg Identification (%)</th>
<th>Argument Typing (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>D</td>
<td>LSTM Joint</td>
<td>59.3</td>
<td>54.3</td>
<td>56.7</td>
<td>55.1</td>
</tr>
<tr>
<td>D</td>
<td>Transfer</td>
<td>55.8</td>
<td>67.4</td>
<td>61.1</td>
<td>50.6</td>
</tr>
<tr>
<td>D</td>
<td>Transfer</td>
<td>85.7</td>
<td>41.2</td>
<td>55.6</td>
<td>75.5</td>
</tr>
</tbody>
</table>

- Errors: misclassification within the same scenario
  - e.g., Being-Born v.s. Giving-Birth

Abby was a true water *birth* (3kg - normal) and with Fiona I was dragged out of the pool after the head crowned.
Discussion

- **Impact of AMR Parsing**
  - AMR is used to identify candidate triggers and arguments, as well as construct event structures
  - Compare AMR with Semantic Role Labeling (SRL) on a subset of ERE corpus with perfect AMR annotations
  - Train on top-6 most popular seen (training) types: *Arrest-Jail, Execute, Die, Meet, Sentence, Charge-Indict*, and test on 200 sentences, with 128 attack event mentions and 40 convict event mentions

<table>
<thead>
<tr>
<th>Method</th>
<th>Trigger Labeling</th>
<th>Argument Labeling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Perfect AMR</td>
<td>79.1</td>
<td>47.1</td>
</tr>
<tr>
<td>Perfect AMR with Core Roles only (SRL)</td>
<td>77.1</td>
<td>47.0</td>
</tr>
<tr>
<td>System AMR</td>
<td>85.7</td>
<td>32.0</td>
</tr>
</tbody>
</table>
Discussion

- Transfer Learning v.s. Supervised Model
  - Target Event Ontology: ACE(33 types)+FrameNet (1161 frames)
  - Seen types for training: 10 most popular ACE types
  - Unseen type: 23 remaining ACE types
Conclusion and Future Work

- We model event extraction as a generic grounding problem, instead of classification.
- By leveraging existing human constructed event schemas and manual annotations for a small set of seen types, the zero shot framework can improve the scalability of event extraction and save human effort.
- In the future, we will extend this framework to other Information Extraction problems.
Q&A

Thank You!