Zero-Shot Transfer Learning for Event Extraction

Lifu Huang[1], Heng Ji[1], Kyunghyun Cho[2], Ido Dagan[3], Sebastian Riedel[4], Clare R. Voss[5]

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

Rensselaer

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

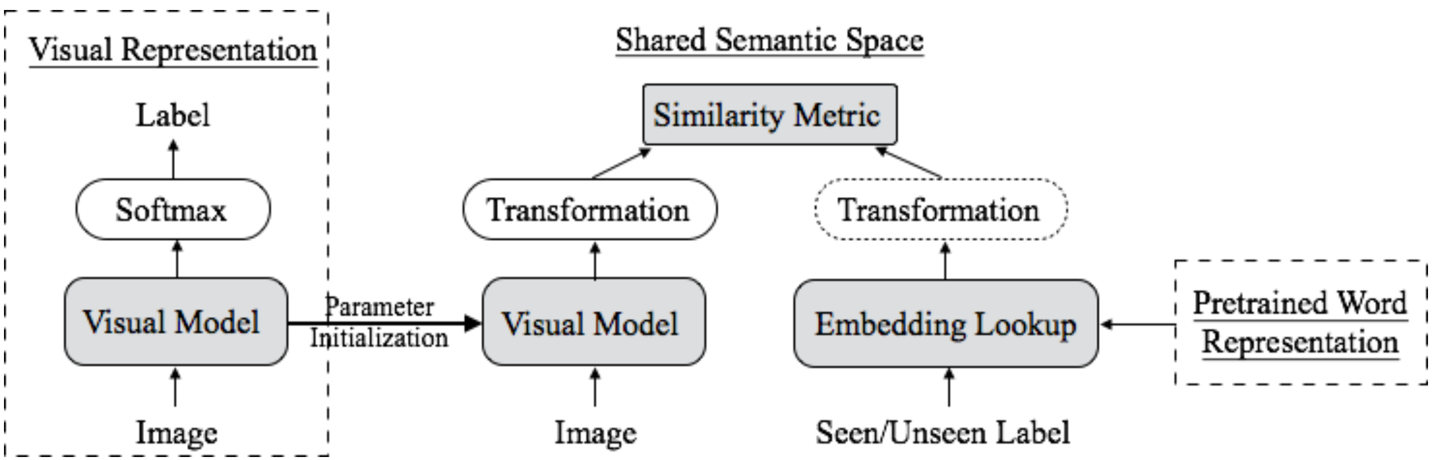
Traditional Event Extraction Pipeline

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
…

*The resources for existing event types **cannot** be re-used for new types; not to mention we have **1000+** event types*

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



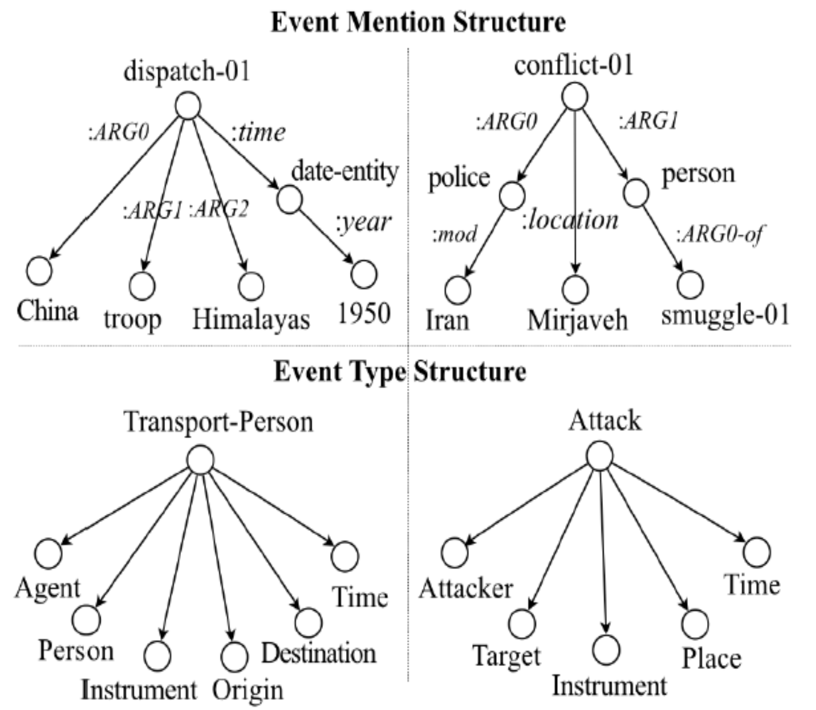
Andrea Frome, Greg S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc Aurelio Ranzato, Tomas Mikolov, DeViSE: A Deep Visual-Semantic Embedding Model

Motivation

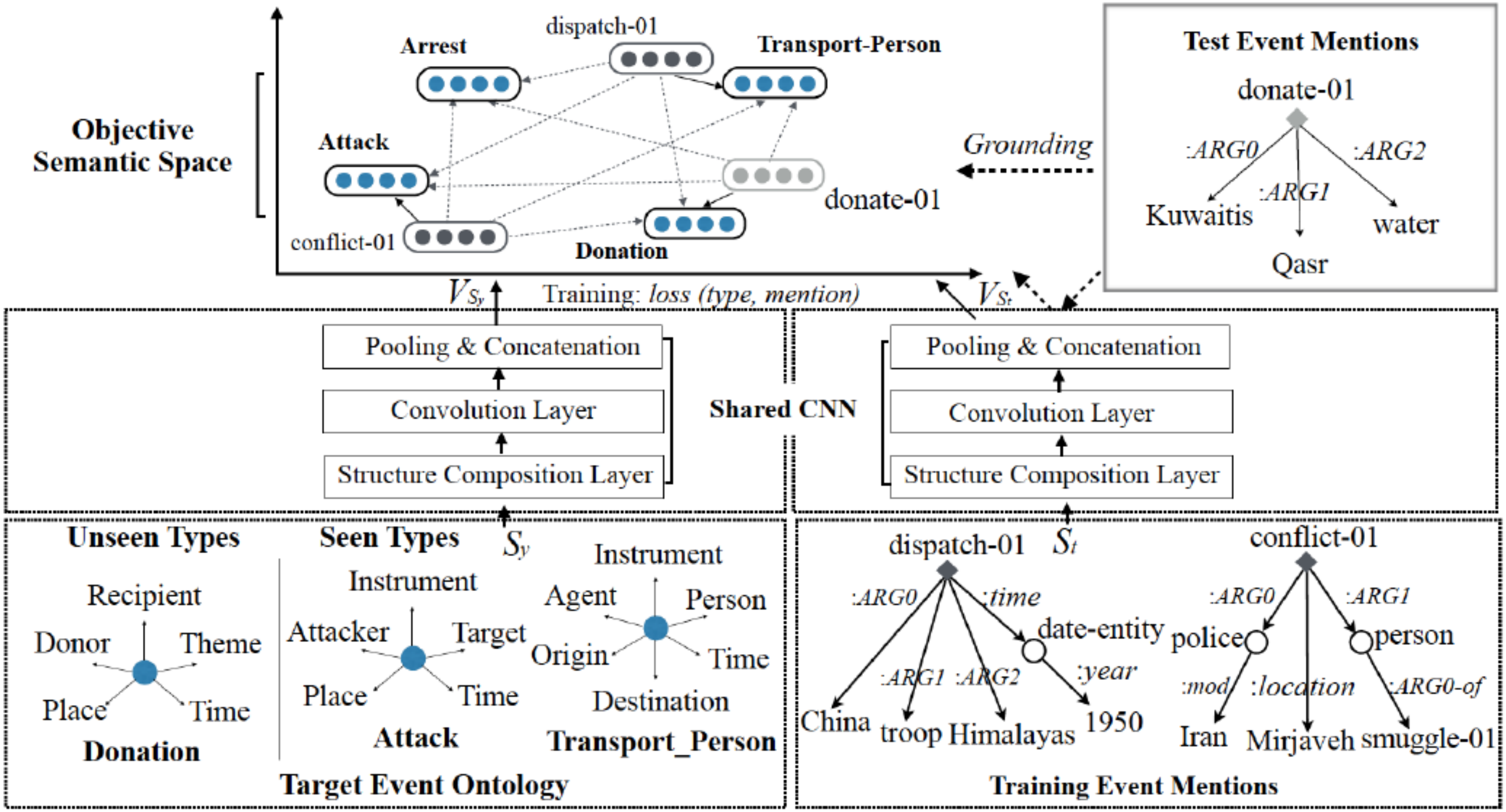
- Zero Shot Learning for Event Extraction
  - both event mentions and types have rich semantics and structures, which can specify their consistency and connections

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

Categories	Relations
Core Roles	ARG0, ARG1, ARG2, ARG3, ARG4
None-Core Roles	mod, location, instrument, poss, manner, topic, medium, prep-X
Temporal	year, duration, decade, weekday, time
Spatial	destination, path, location

- Event and Type Structure Construction

Approach Details

- Structure Composition and Representation
  - Event Mention Structure
    - We use a matrix M_λ to represent each AMR relation λ, and compose its semantics with two concepts for each tuple:

      $u = \langle w_1, \lambda, w_2 \rangle$ e.g., *<dispatch-01, :ARG0, China>*

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

  - 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

      $u' = \langle y, r \rangle$ e.g., *<Transport_Person, Person>*

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

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

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

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

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

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

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 *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

Setting	Top-N	Seen Types for Training/Dev
A	1	Attack
B	3	Attack, Transport, Die
C	5	Attack, Transport, Die, Meet, Arrest-Jail
D	10	Attack, Transport, Die, Meet, Arrest-Jail, Transfer-Money, Sentence, Elect, Transfer-Ownership, End-Position

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)

Setting Index	Training			Development		Test		
	# of Types/Roles	# of Events	# of Arguments	# of Events	# of Arguments	# of Types/Roles	# of Events	# of Arguments
A	1/5	953/900	894/1,097	105/105	86/130	23/59	753	879
B	3/14	1,803/1,500	2,035/1,791	200/200	191/237			
C	5/18	2,033/1,300	2,281/1,503	225/225	233/241			
D	10/37	2537/700	2,816/879	281/281	322/365			

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

Method	Hit@k Trigger Typing (%)			Hit@k Argument Typing (%)		
	1	3	5	1	3	5
WSD-Embedding	1.73	13.01	22.84	2.39	2.84	2.84
Transfer A	3.98	23.77	32.54	1.25	3.41	3.64
Transfer B	7.04	12.48	36.79	3.53	6.03	6.26
Transfer C	20.05	34.66	46.48	9.56	14.68	15.70
Transfer D	33.47	51.40	68.26	14.68	26.51	27.65

Evaluation

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

Type	Subtype	Hit@k Trigger Typing%		
		1	3	5
Justice	Sentence	68.29	68.29	69.51
Justice	Appeal	67.50	97.50	97.50
Justice	Release-Parole	73.91	73.91	73.91
Conflict	Attack	26.47	44.52	46.69
Transaction	Transfer-Money	48.36	68.85	79.51
Business	Start-Org	0	33.33	66.67
Movement	Transport	2.60	3.71	7.81
Personell	End-Position	9.09	50.41	53.72
Life	Injure	87.64	91.01	91.01
Contact	Phone-Write	60.78	88.24	90.20

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

Setting	Method	Trigger Identification (%)			Trigger Typing (%)			Arg Identification (%)			Argument Typing (%)		
		P	R	F	P	R	F	P	R	F	P	R	F
D	LSTM	59.3	54.3	56.7	55.1	50.4	52.6	47.8	22.6	30.6	28.9	13.7	18.6
D	Joint	55.8	67.4	61.1	50.6	61.2	55.4	36.4	28.1	31.7	33.3	25.7	29.0
D	Transfer	85.7	41.2	55.6	75.5	36.3	49.1	28.2	27.3	27.8	16.1	15.6	15.8

  - 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.

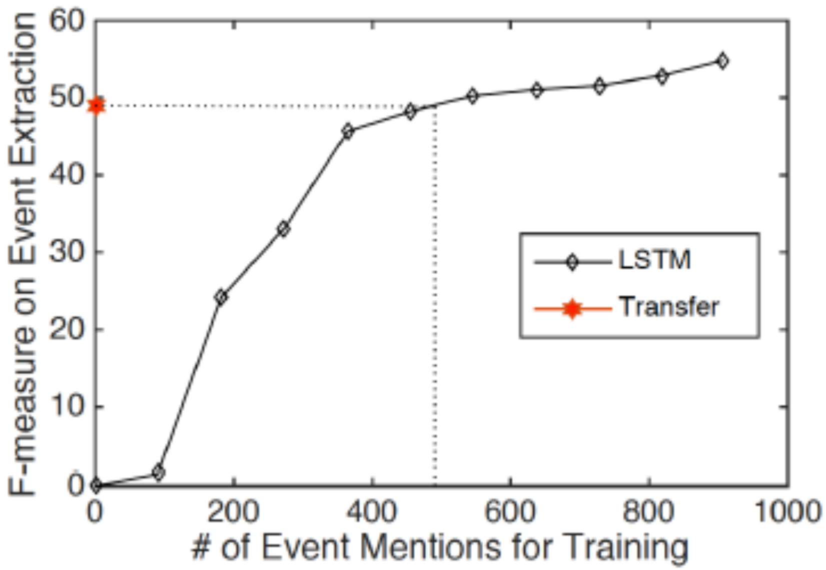
Discussion

- 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

Method	Trigger Labeling			Argument Labeling		
	P	R	F_1	P	R	F_1
Perfect AMR	79.1	47.1	59.1	25.4	21.4	23.2
Perfect AMR with Core Roles only (SRL)	77.1	47.0	58.4	19.7	16.9	18.2
System AMR	85.7	32.0	46.7	22.6	15.8	18.6

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!