What Happens Next? Event Prediction Using a Compositional Neural Network Model

Mark Granroth-Wilding, Stephen Clark

AAAI, 2016

Soham Parikh (sohamp@seas.upenn.edu)

2/20/19

Penn Engineering

What is an Event?

Predicate

Predicate

x_0 x_0 x_0

Text: Robbers made a big score, fleeing after stealing more than \$300,000 from Wells Fargo armored-truck guards who were servicing the track's ATMs, the Police Department said. The two Wells Fargo guards reported they were starting to put money in the clubhouse ATM when a man with a gun approached and ordered them to lie down. . .

x_0

Entity mentions: {*Wells Fargo armored-truck guards*, *The two Wells Fargo guards*, *they*, . . .}

Predicate events: service(x_0, ATMs), report(x_0), put(x_0, money, in clubhouse), lie+down(x_0), . . .

C&J08 events: *(service, subj)*, *(report, subj)*, *(put, subj)*, *(lie+down, subj)*

Predicate

Predicate

- **Event** : said to be described each time an entity is an argument to a verb (assumption)
- **Representation**: verb(subject, object (optional))
  - e.g. eat(John, spaghetti)

Problem Overview

- **Narrative chain**: Partially ordered set of events sharing a common entity
- **Event Prediction**: Predict missing events in a narrative chain
- Why is it important?
  - Requires good understanding of event descriptions
  - Requires good representation of events

EXAMPLE:

- Context Events:

  play(x, tennis), enter(x, tournament), win(x, final)

- Options:
  1. lift(x, trophy)
  2. cook(x, spaghetti)
  3. kill(x, spider)
  4. discover(x, truth)
  5. drive(x, car)

Problem Overview

- Narrative Cloze Task:
  - Sequence of events with a missing event
  - Need to predict the missing event given the rest
  - Cons: Large number of possibilities
- Multiple Choice Narrative Cloze:
  - Input: A sequence of events, 5 candidate events
  - Output: A candidate event
  - Better performance measure (Accuracy)

EXAMPLE:

- Context Events:

  play(x, tennis), enter(x, tournament), win(x, final)
- Options:
  1. lift(x, trophy)
  2. cook(x, spaghetti)
  3. kill(x, spider)
  4. discover(x, truth)
  5. drive(x, car)

Contents:

Previous approaches

- Want measure of relevance between context and candidate event
- Relevance measured as point-wise mutual information between context and candidate events [Chambers and Jurafsky, '08]
- Relevance of event measured using bigram probability in terms of events [Jans et al, '12]

- Probabilities are estimated using counts (and smoothing) from training corpus

$$s(c) = \sum_{i=0}^{n-1} ppmi(e_i, e_c)$$

$$ppmi(x, y) = max\left(log_2\left(\frac{P(x,y)}{P(x)P(y)}\right), 0\right)$$

$$s(c) = \frac{1}{n}\sum_{i=0}^{n-1} P(e_c \mid e_i)$$

Dataset

- Events extracted from the NYT articles of Gigaword corpus
- PoS tagging and dependency parsing (using C&C tools) for identifying verb, subject, object. Verbs lemmatized.
- Coreference resolution using OpenNLP
- Predicative adjectives for the verbs "be" and "become":
  - e.g., X was upset → be(X, upset)
- Remove events with high frequency and low meaning
- Incorrect options randomly sampled from other chains

Entities

x_0 = Giardino x_1 = chairman, him

Context (e_i)

die(x_0), attend(x_0, reunion), specialize(x_0, as partner), describe(x_0, x_1, as product), hold(x_0, position), appoint(–, x_0, to the board), lead(x_0, effort), improve(x_0, operation), propose(x_0, cut), play(x_0, role), ____

c_1: receive(x_0, response)

c_2: drive(x_0, mile)

c_3: seem(x_0)

c_4: discover(x_0, truth)

c_5: modernize(x_0, procedure)

?

Answer: c_5

Motivation

- **Aim**: Predict next event
- **How**: Measure relevance between context and candidate events
- Verbs which occur in similar contexts are more relevant

  e.g., *diving* and *swimming* as opposed to *diving* and *talking*
- **Count-based methods**: high probability only to events occurring in training corpus
- **Word embeddings**: Map words to fixed-length vectors. Expectations:
  - Similar words have "similar" vectors
  - Good semantic properties
  - Capture relation between words not seen together while training

  e.g., criticize(politician, law), repeal(parliament, law)
- ∴ Use word embeddings to represent events

Models using external knowledge

- **Mikolov-Verb**:
  - Represent events using pre-trained word embedding for its verb
  - Relevance score obtained from cosine similarity between candidate event and sum of context events vectors
- **Mikolov-Verb-Arg**:
  - Arguments (subject, object) contain information as well
  - Represent events as sum of pre-trained word embeddings of verbs and arguments
  - Relevance score is cosine similarity as before

Models trained on corpus

Predicate events: service(x_0, machine), report(x_0), put(x_0, money, in clubhouse), lie+down(x_0), . . .

a. `word2vec` 'sentence': *service:subj report:subj put:subj lie+down:subj*

b. `word2vec` 'sentence' with arguments: *service:subj arg:guards arg:machine report:subj arg:guards put:subj arg:guards arg:money arg:clubhouse lie+down:subj arg:guards*

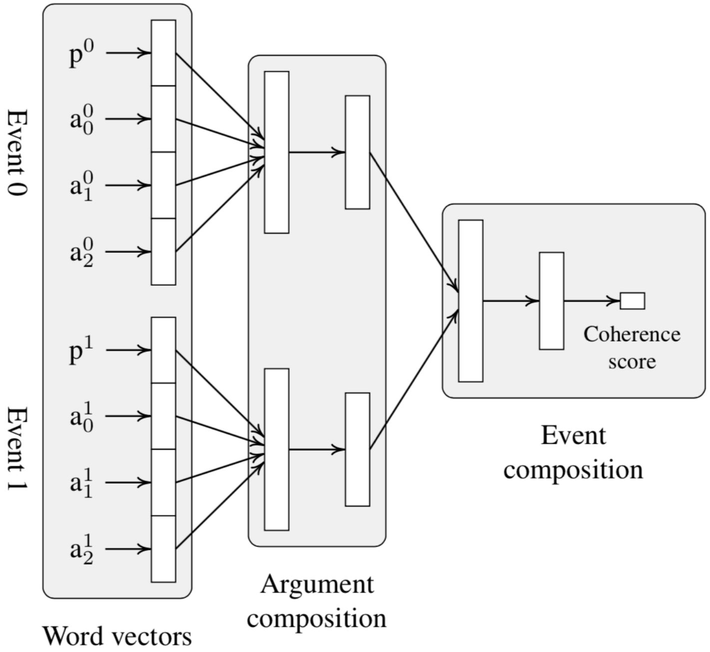
- **Word2Vec-Pred**:
  - Learn word embeddings instead of using pre-trained embeddings
  - Represent each event as a single word
  - Train a skip-gram model to get event embeddings
  - **Score**: Cosine Similarity

- **Word2Vec-Pred+Arg**:
  - Use information from arguments
  - Treat verb and arguments as separate words
  - Words from a single narrative chain form a sentence
  - Skip-gram model to get words embeddings
  - Event: Sum of argument and predicate word embeddings
  - **Score**: Cosine Similarity

Compositional Model

- **Event-Comp**:
  - Need a better event representation
  - Obtain event representation by non-linear combination of embeddings using feedforward neural network
  - Initialize predicate and argument embeddings as in **Word2Vec-Pred+Arg**
  - **Score**: Feed-forward network to obtain a coherence score (scalar) between two events

Event-Comp



Type to enter a caption.

- **Objective Function:**

$$min_{\theta}\left(-\sum log(Score) + \lambda L(\theta)\right)$$

$$Score = \quad 1[e_0 = e_1]coh(e_0, e_1) + 1[e_0 \neq e_1](1 - coh(e_0, e_1))$$

$coh(e_0, e_1)$: Coherence score

$L(\theta)$: Regularization Term

Experiment Details

- **Evaluation**: Accuracy
- **Event-Comp**: Positive examples from same chain. Negative from other chains with entity replaced
- 300-dimensional word embeddings
- > 830k documents. >11 million event chains.
- 10% for development set, 10% for test set.

Entities

x_0 = Giardino x_1 = chairman, him

Context (e_i)

die(x_0), attend(x_0, reunion), specialize(x_0, as partner), describe(x_0, x_1, as product), hold(x_0, position), appoint(–, x_0, to the board), lead(x_0, effort), improve(x_0, operation), propose(x_0, cut), play(x_0, role), ____

c_1: receive(x_0, response)

c_2: drive(x_0, mile)

c_3: seem(x_0)

c_4: discover(x_0, truth)

c_5: modernize(x_0, procedure)

?

Answer: c_5

Result & Analysis

System	Accuracy(%)
Chance Baseline	20.00
C&J08	30.52
BIGRAM	29.67
DIST-VECS (using LSI)	27.94
MIKOLOV-VERB	24.57
MIKOLOV-VERB+ARG	28.97
Word2Vec-Pred	40.17
Word2Vec-Pred+Arg	42.23
EVENT-COMP	49.57

- C&J08 performs relatively better than a lot of models
- Learning word embeddings using predicates from event chains improves accuracy by a margin
- Including argument embeddings enhances performance
- Using a non-linear combination for the representation of an event performs better than a linear combination of events

Conclusions

- Better task in terms of evaluation of performance of models
- Skip-gram or CBOW can be used to get event representations
- Arguments are important while considering events

Shortcomings

- Incomplete information about events
- Not all events are included in the chain
- The sequence of events is not taken into account
- Does not prevent model from making inconsistent/contradictory judgments
- No error analysis for where the model makes mistakes
- No comparison b/w coherence score and cosine similarity

EXAMPLE:

- Context Events:

  participate(x, race), run(x), lead(x, race), fall(x)
- Options:
  1. win(x)
  2. injure(x)

Future Work

- Use temporal/sequence information
- Apply constraints to deal with inconsistent/contradictory results
- Combine an entire chain of events instead of considering pairs of events
- Better event representation
- Event prediction/generation using unstructured text
- Extend this model to story/event generation

Thank You!