Integrating Order Information and Event Relation for Script Event Prediction

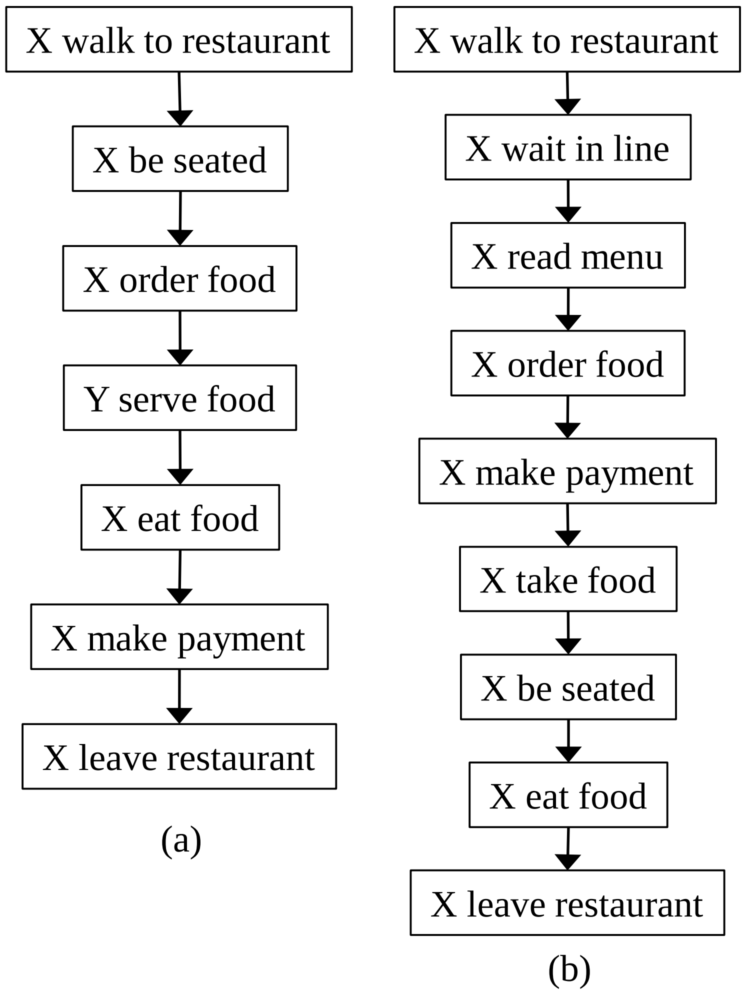
Zhongqing Wang[1,2], Yue Zhang[2] and Ching-Yun Chang[2]

[1]*Soochow University, China*

[2]*Singapore University of Technology and Design*

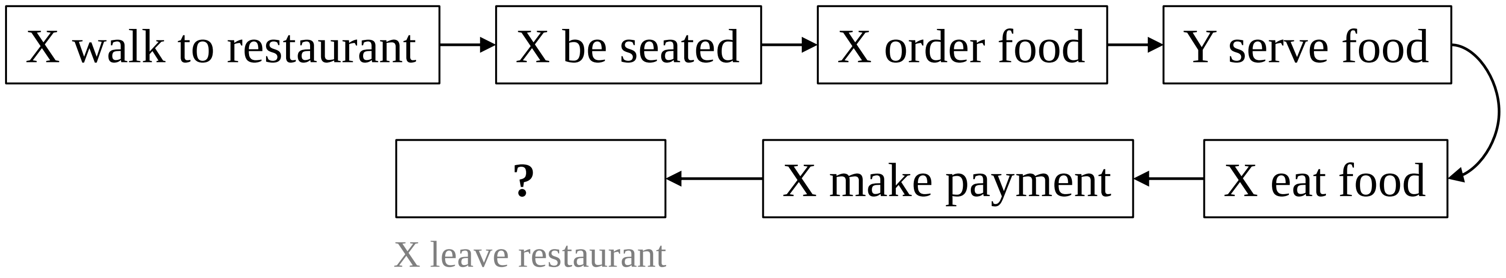
Event Chain

- Frequently recurring sequences of events in prototypical scenarios, are a useful source of world knowledge.



New Event Prediction

- **Narrative Cloze Test.** Asks for *a missing event* in a given event chain with a gap.



- Drawback: Narrative cloze test is that there can sometimes be multiple plausible answers, but only one gold-standard answer
  - which can make it overly expensive to manually evaluate system outputs.

Multi-Choice Narrative Cloze (MCNC)

- Aim to choose the most likely next event from a set of candidates, given a chain of events.

Entities

X = Customer, Y = Waiter

Context(e_i)

walk(X, restaurant), seat(X), order(X, food), serve(Y, food) eat(X, food), make(X, payment), __________

c_1: receive(X, response)
c_2: drive(X, mile)
c_3: seem(X)
c_4: discover(X, truth)
c_5: leave(X, restaurant)

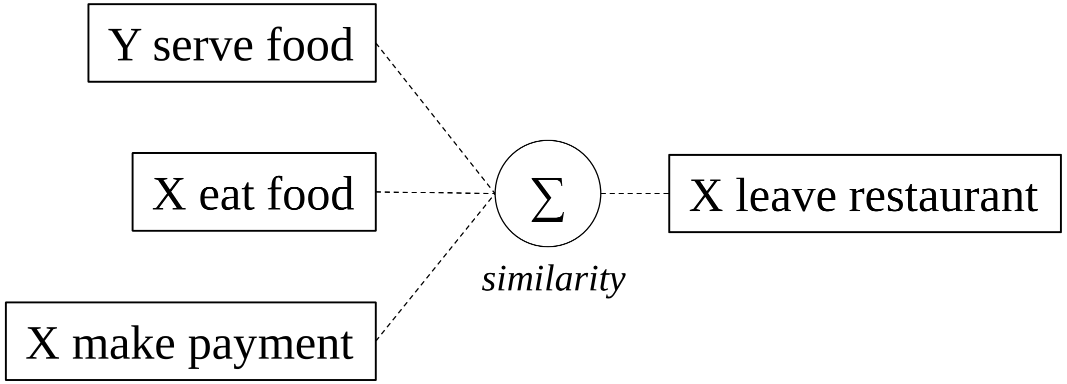
?

Predict Next Event for MCNC

- Pair-wise model
- Sequence Model

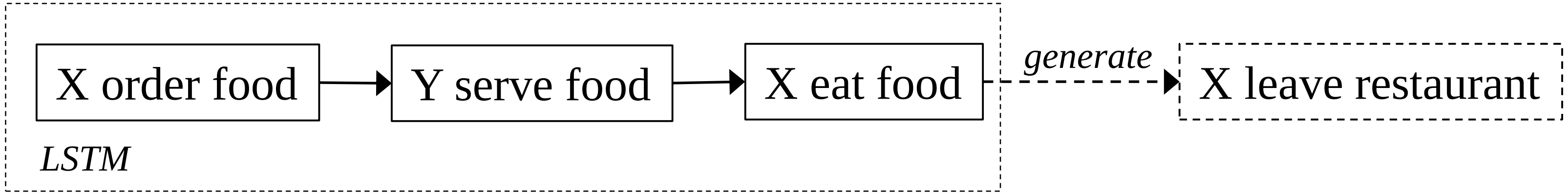
Pair-wise Event Prediction

- Induce event chains by grouping events based on their narrative coherence
  - PMI (Chambers and Jurafsky, 2008)
  - Skip Bi-gram Probabilities (Jans et al., 2012)
  - Log-bilinear neural language model (Rudinger et al., 2015)
  - Siamese Network (Granroth-Wilding and Clark, 2016)



Sequence based Event Prediction

- Pichotta and Mooney (2016) experimented with LSTM for script learning, using an existing sequence of events to predict the probability of a next event.
  - LSTMs can encode unbounded time sequences without losing long-term historical information.
  - LSTMs capture significantly more order information compared to the methods.



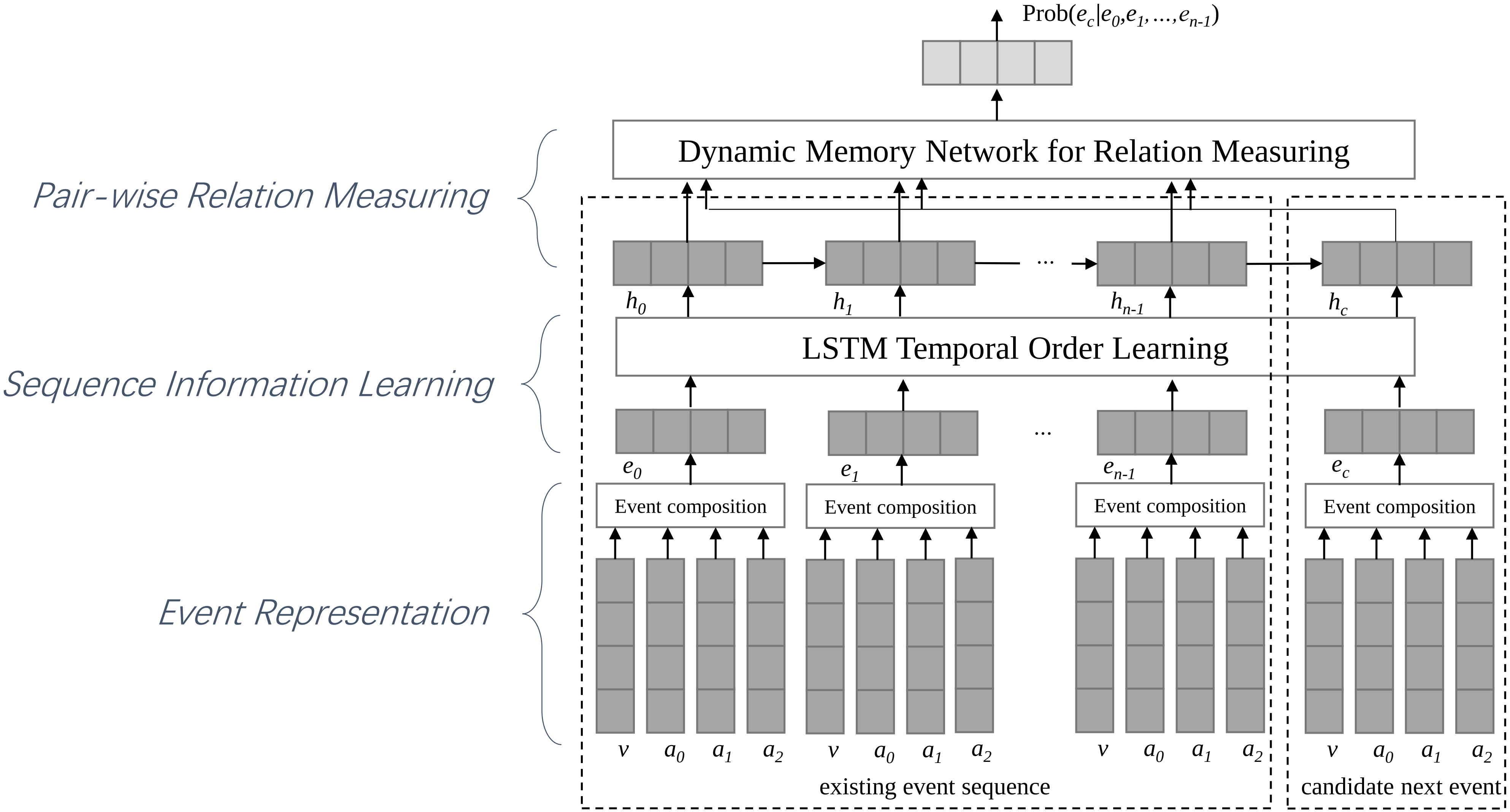
Sequence based Event Prediction (cont.)

- A strong-order LSTM model can also suffer the disadvantage of over-fitting, given the flexible order of event chains in a script.
- Event-pair models are more adaptive for flexible orders.
- **Challenge:** *How to integrate both sequence and pair-wise model?*

Model

- We calculate event pair relations by representing events in a chain using LSTM hidden states, which encode temporal information.
  - We use the *temporal-order* in a chain as a feature for event pair modeling.
  - we use a *dynamic memory network* model to automatically induce event weights for each event for inferring the next event.

Model (cont.)



Prob($e_c|e_0,e_1,\dots,e_{n-1}$)
Dynamic Memory Network for Relation Measuring
Pair-wise Relation Measuring
h_0
h_1
...
h_{n-1}
h_c
LSTM Temporal Order Learning
Sequence Information Learning
e_0
e_1
...
e_{n-1}
e_c
Event composition
Event composition
Event composition
Event composition
Event Representation
v a_0 a_1 a_2
v a_0 a_1 a_2
v a_0 a_1 a_2
v a_0 a_1 a_2
existing event sequence
candidate next event

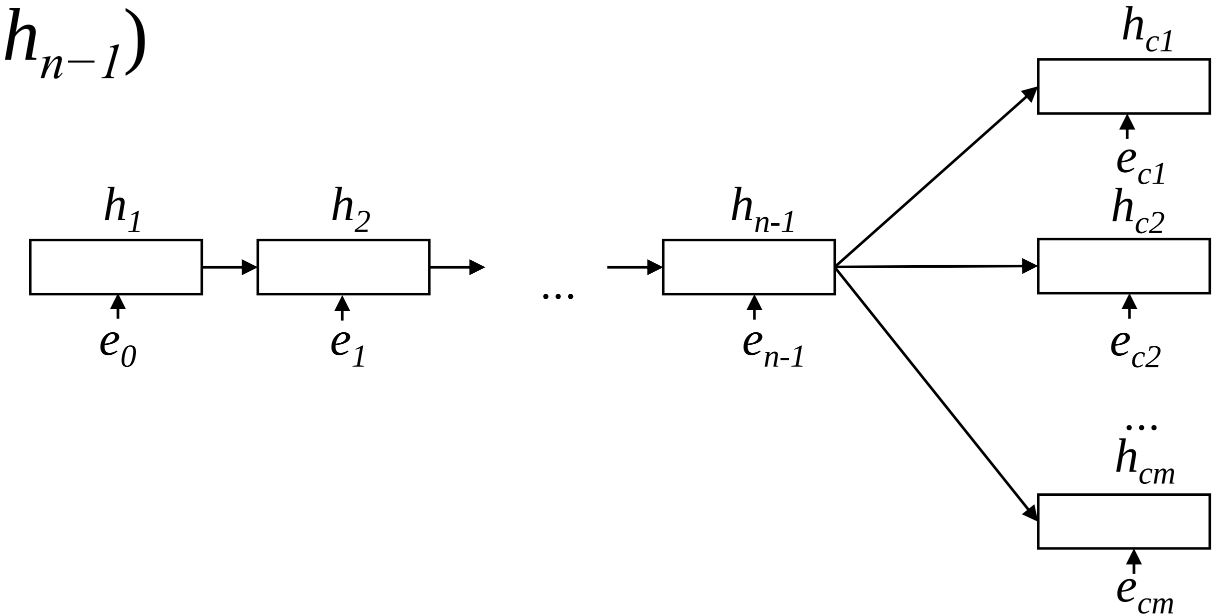
Event Representation

- For a event e=(v, a_0, a_1, a_2)
  - We learn vector representations of a event e by composing pre-trained word embeddings of its verb (v) and arguments (a_0, a_1, a_2)
  - Denoting the embeddings of v, a_0, a_1, and a_2 as $e(v)$, $e(a_0)$, $e(a_1)$, and $e(a_2)$, respectively, the embedding of e is calculated using a **tanh** composition layer

$$e(e) = \tanh(W_e^v \cdot e(v) + W_e^0 \cdot e(a_0) + \\ W_e^1 \cdot e(a_1) + W_e^2 \cdot e(a_2) + b_e)$$

Modeling Temporal Orders

- **Existing events in the chain.** We obtain a sequence of hidden state vectors h_1, h_2, ..., h_{n-1} by recurrently feeding $e(e_1)$, $e(e_2)$, ..., $e(e_{n-1})$ as inputs to the LSTM, where $h_i = \mathrm{LSTM}(e(e_i), h_{i-1})$.
- **Candidate next event.** $e(e_c)$ is appended to the existing event chain to obtain a temporal-order sensitive feature vector h_c, where $h_c = \mathrm{LSTM}(e(e_c), h_{n-1})$



Modeling Pairwise Event Relations

- Given a pair of events $h_i (i \in [1..n-1])$ and h_c, the relatedness score is calculated by

$$s_i = \mathrm{sigmoid}(W_{si} h_i + W_{sc} h_c + b_s),$$

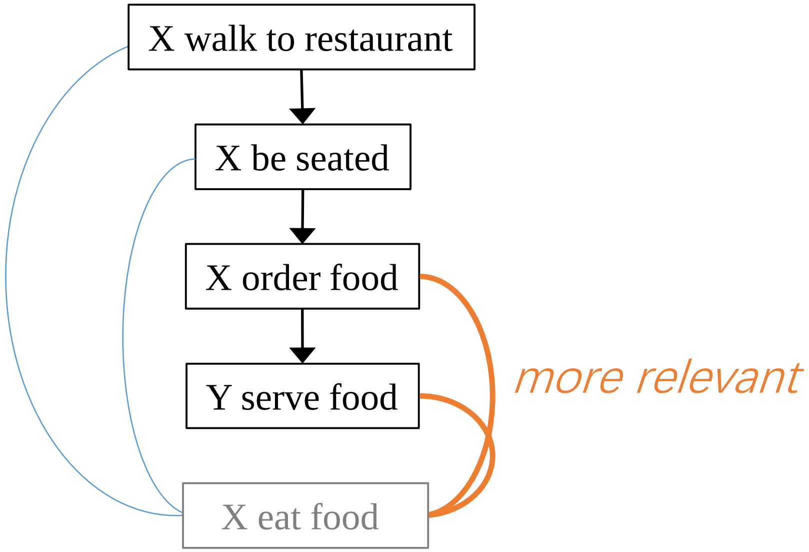
- Given the relation score s_i between h_c and each existing event h_i, the likelihood of e_c given e_1, e_2, ..., e_{n-1} can be calculated as the average of s_i

$$s = \frac{\sum_{i=1}^{n-1} \boxed{s_i}}{n-1}$$

relation score between h_c and h_i

Modeling Pairwise Event Relations (cont.)

- The drawback is that it considers the contribution of each event on the chain is same.
- However, given a chain of existing events, some are more informative for inferring a subsequent event than others.



Weighting Existing Events

- We use an attentional neural network to calculate the relative importance of each existing event according to the subsequent event candidate

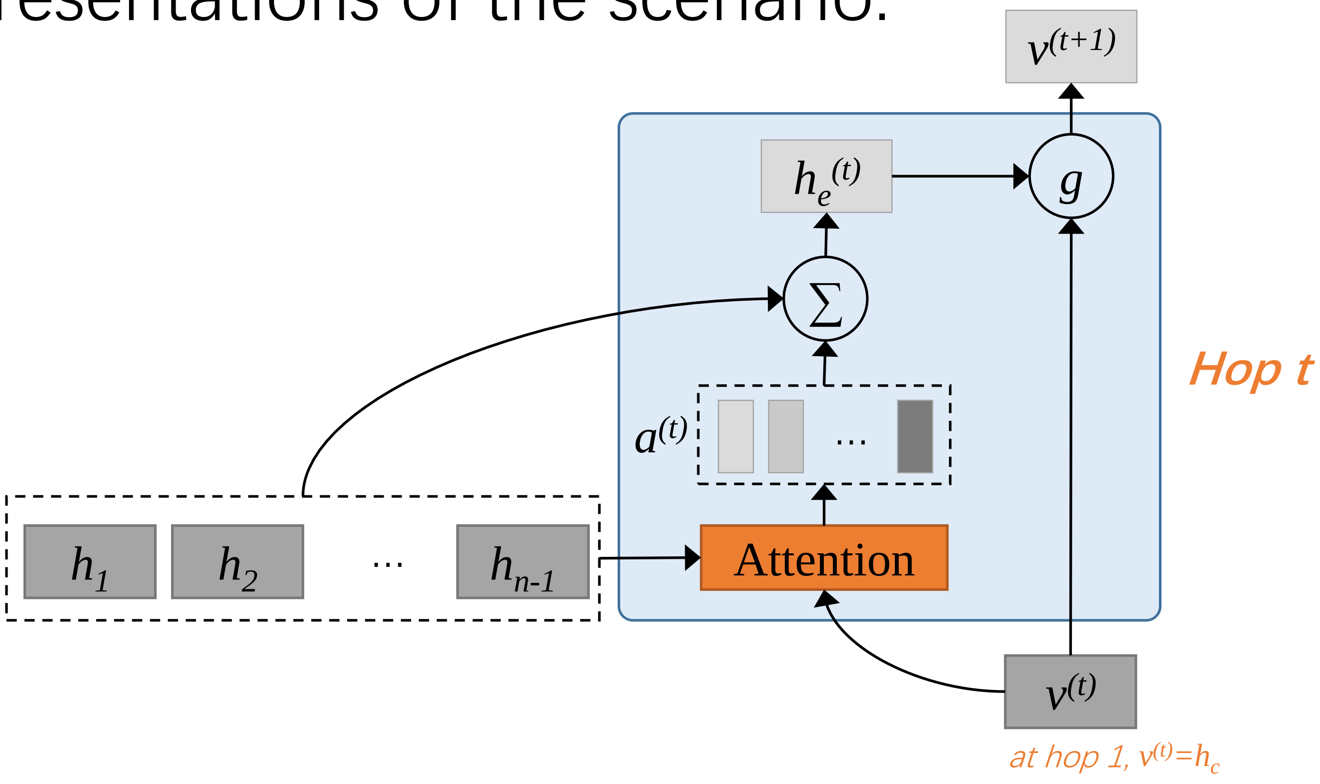
[redacted]

$$\alpha_i = \frac{\exp(u_i)}{\sum_j \exp(u_j)}$$

$$s = \sum_{i=1}^{n-1} \underbrace{\boxed{\alpha_i}}_{\text{weight of event } i} \cdot \boxed{s_i} \quad \text{relation score between } h_c \text{ and } h_i$$

Multi-layer Attention using Deep Memory Network

- We use a *deep memory network model* to refine event weight and event relation calculation by recurrently modeling more abstract representations of the scenario.



Training

- Our training objective is to minimize the cross-entropy loss between the gold subsequent event and the set of non-subsequent events. The loss function of event chain prediction is that:

$$L(\Theta) = \sum_{i=1}^{N} (s_i - y_i)^2 + \frac{\lambda}{2} ||\Theta||^2$$

Datasets

- Following Granroth-Wilding and Clark (2016), we extract events from the NYT portion of the Gigaword corpus.
- The training set consists of 1,500,000 event chains.
- We use 10,000 event chains as the test set, and 1,000 event chains for development.

Influence of Event Structure

- The results demonstrates the central value of the verb in denoting a event
- it also suggests that the arguments themselves play a useful role in inferring the stereotypical scenario

Method	**Acc. (%)**
MemNet	**54.36**
[redacted]	

Influence of Network Configurations

- Temporal order information over the whole event chain does have significant influence on the results
- Giving different weights to different events does lead to improving results

Method	**Acc. (%)**
MemNet	**54.36**
-Hop	52.03
-Attention	50.76
-LSTM	51.72
-Hop,-LSTM	50.65
-Attention,-LSTM	48.26
LSTM-Only	46.72

Thanks

wangzq.antony@gmail.com, yue_zhang@sutd.edu.sg,
chang.frannie@gmail.com