Context-Aware Neural Model for Temporal Information Extraction

Yuanliang Meng and Anna Rumshisky
Text Machine Lab, Department of Computer Science, University of Massachusetts Lowell

Overview

- "*Can we mimic human ability to integrate information from previously processed context to understand the order and timing of events in a narrative?*"
- We propose the **Global Context Layer (GCL)** to process and retrieve information about previously processed context.
- Inspired by the Neural Turing Machine (NTM), GCL has **long-term memory** and soft **attention** addressing, and thus can **resolve long-distance dependencies**.
- It has a uniform architecture for event-event, event-timex and timex-timex pairs, so there is no need to train separate models.

Our repository is available: https://github.com/text-machine-lab/TEA

Dataset

We used the Timebank-Dense data (https://www.usna.edu/Users/cs/nchamber/caevo/#corpus)
Training files are annotated with event tags, temporal expression tags (timexes) and temporal relation tags (TLINKs).

Director of the U.S. Federal Bureau of Investigation (FBI) Louis Freeh said here Friday that U.S. air raid on Afghanistan and Sudan is not directly linked with the probe into the August 7 bombings in east Africa.

AFTER IS_INCLUDED

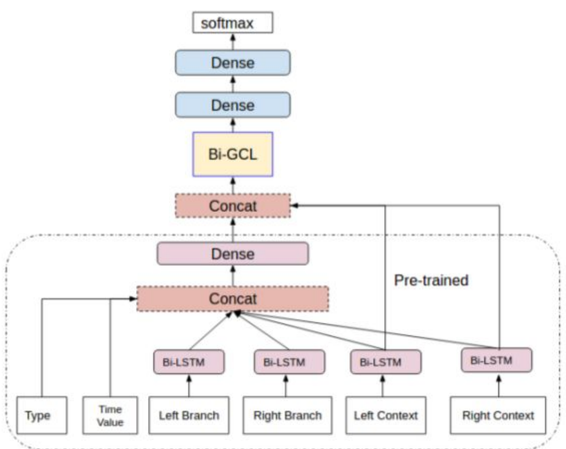
All the possible temporal relations within a sentence or cross consecutive sentences are labeled (so it is called “dense”).
There are 22 training files, 5 validation files and 9 test files.

Experiment Setup

Input Features

- Pair type: event-event, event-timex, or timex-timex
- Time value: for timex-timex pairs only. A tuple of real values indicating the difference.
  - For <TIMEX3 tid="t1" type="DATE" value="2018-08-21">Friday</TIMEX3> and <TIMEX3 tid="t2" type="DATE" value="1998-08">August</TIMEX3>,
    t2 := (2018 + 7/12+ 21/365, 2018 + 7/12 + 7/365) = (2018.64, 2018.64)
    t3 := (2018 +7/12, 2018 + 7/12 + 31/365 = (2018.58, 2018.67)
    Input = t2 - t3 = (0.06, -0.03) *This represents IS_INCLUDED relation*
- Words on syntactic dependency path:
  - Intra-sentence pairs: the shortest path between the two entities of interest.
  - Cross-sentence pairs: the paths between entities and their sentence roots, respectively.
- Words in context: The entity mentions and their surrounding words.

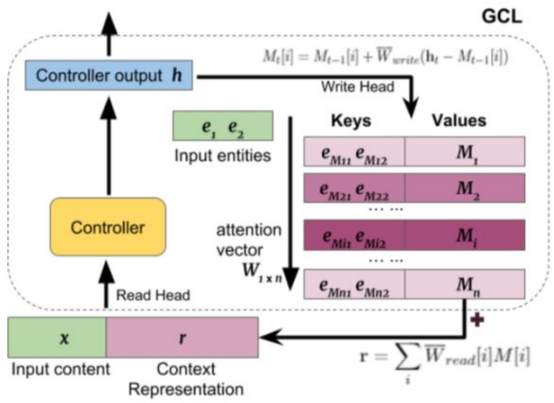
Word tokens are initialized with glove.840B.300d word vectors (300 dimensional).



Two models

We trained a strictly pairwise model first, without Global Context Layer. Then concatenate the hidden layer of the trained model with the GCL and an output layer.

Global Context Layer



- GCL has a **controller**, a **memory**, some **read head(s)**, and some **write head(s)**.
- Before an entity pair is processed, the controller reads context information from the memory.
- After a pair is processed, the new information is saved in memory to serve as context.
- The memory has two components: address part and content part. It is dynamically updated, until all pairs are processed.

The address part of GCL memory is a concatenation of two entity representations.

$$K[i] = \mathbf{e_{M1}}[i] \oplus \mathbf{e_{M2}}[i]$$

When a new entity pair comes, it is compared with the memory addresses to compute attention weight.

$$D[i] = \frac{1}{Z}||\mathbf{e_1} \oplus \mathbf{e_2} - \mathbf{e_{M1}}[i] \oplus \mathbf{e_{M2}}[i]||_2^2$$

$$D'[i] = \frac{1}{Z'}||\mathbf{e_2} \oplus \mathbf{e_1} - \mathbf{e_{M1}}[i] \oplus \mathbf{e_{M2}}[i]||_2^2$$

$$W[i] = \max(\text{softmax}(\mathbf{1} - \mathbf{D})[i], \text{softmax}(\mathbf{1} - \mathbf{D}')[i])$$

This attention is sharpened, with parameter β:

$$\overline{W}_{read} = \text{softmax}(W^{\beta})$$

β is computed each time:

$$\beta_t = ReLU(W_{sharp}[\mathbf{x}_t, \mathbf{h}_{t-1}] + b_{sharp}) + c_{\beta}$$

The context is read as a weighted sum:

$$\mathbf{r} = \sum_i \overline{W}_{read}[i]M[i]$$

The controller uses input and context to compute output *h* and sends it to succeeding layers. Also the GCL memory is updated.

$$M_t[i] = M_{t-1}[i] + \overline{W}_{write}[i](\mathbf{h}_t - M_{t-1}[i])$$

$$K_t[i] = K_{t-1}[i] + \overline{W}_{write}[i](\mathbf{e_1} \oplus \mathbf{e_2} - K_{t-1}[i])$$

Computing write attention/address is similar to computing read attention, but we use a shift kernel to shift the weights/addresses:

$$\widetilde{W}[i] = \sum_{j=0}^{n-1} W[j]\mathbf{s}[i-j]$$

This shift kernel *s* derives from shift weights C_t

$$C_t = \text{softmax}(W_s[\mathbf{x}_t, \mathbf{h}_t] + b_s)$$

Results

Model	Micro-F1	Macro-F1
CAEVO (not NN model)	.507	
CATENA (not NN model)	.511	
Cheng et al. 2017	.520	
Meng et al. 2017		.519
pairwise	.535	.528
Two more hidden layers	.539	.532
GCL w/ state-tracking controller	.545	**.538**
GCL w/ stateless controller	**.546**	**.538**
GCL w/ pre-trained output layer	.541	.536

- The results in the lower blocks all use **double-check**.
  - Classify a pair in two ways and pick the result with higher score.
- “Two more hidden layers” means adding two hidden layers on top of the pre-trained model without using GCL. It is used as a baseline model.
- The last row corresponds to connecting the output layer (instead of hidden layer) of a pre-trained model to GCL layers with stateless controller.