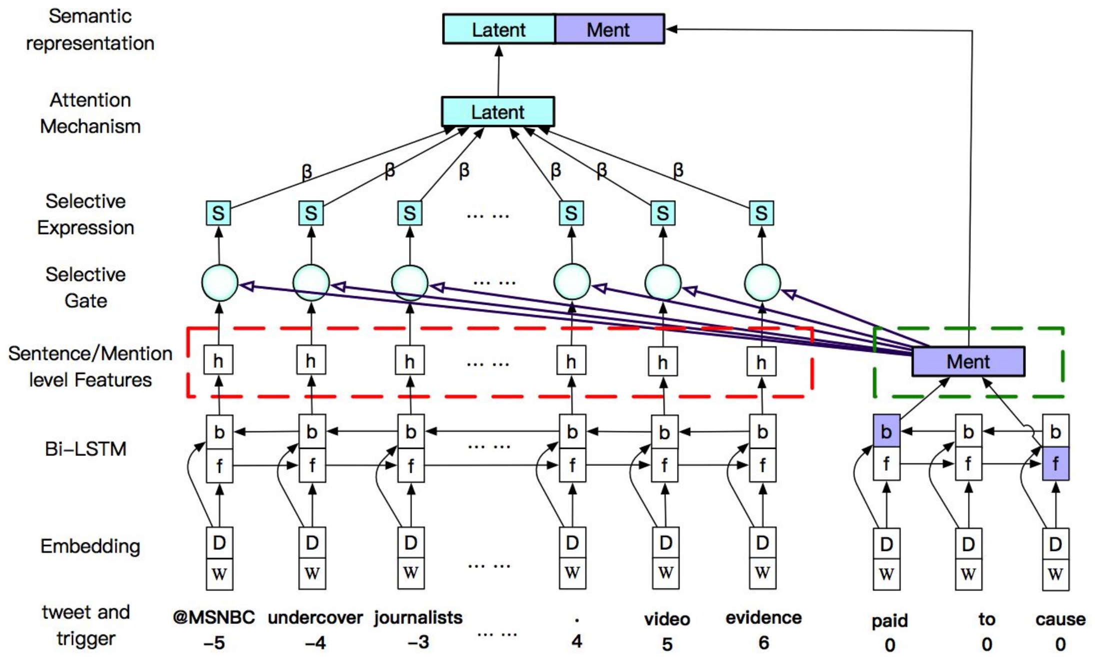
Aim

Goal of this project is to implement the Event Coreference algorithm proposed by Chao et al. in the paper "Selective Expression for Event Coreference Resolution on Twitter".

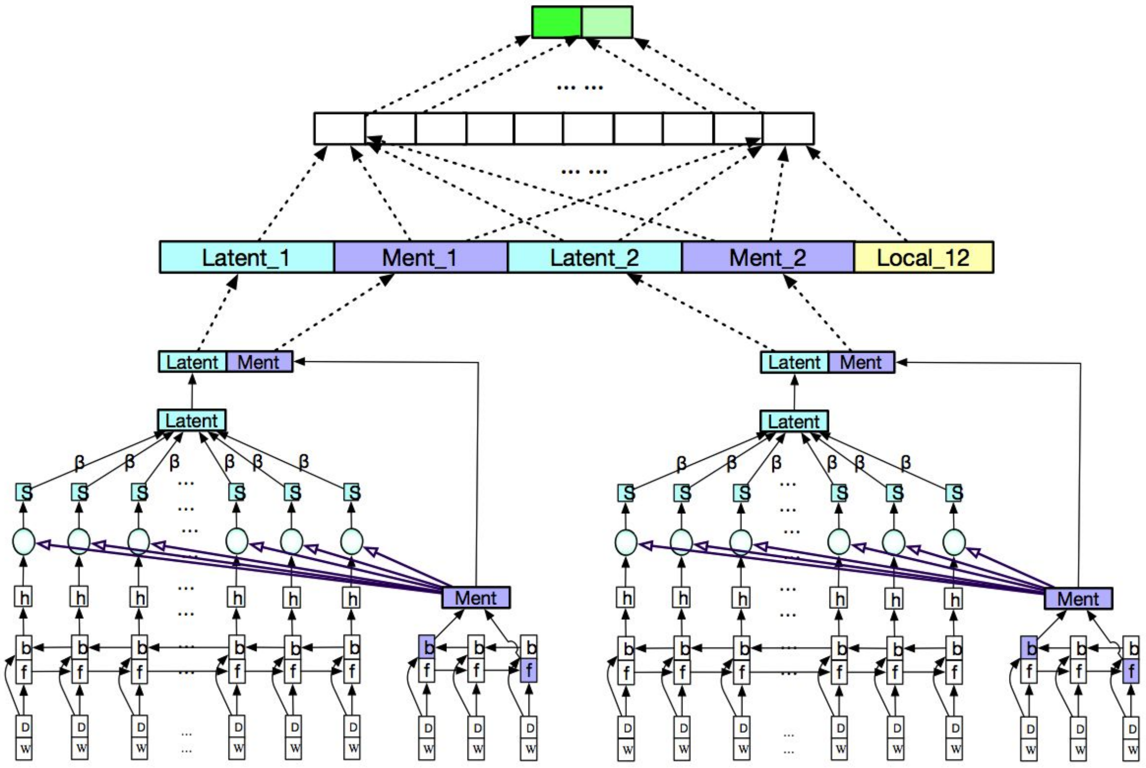
Abstract

Event coreference is the problem of identifying and connecting mentions of the same events in different contexts. It is a fundamental problem in NLP with wide-spread applications.The given paper is the state-of-the-art for event coreference in Twitter. With the growth in popularity and size of social media, there is an urgent need for systems that can recognize the coreference relation between two event mentions in texts from social media. Approach till now basically depend upon NLP features which restricts domain scalability and leads to propagation error.In this paper a novel selective expression approach based on event trigger to explore the coreferential relationship in high-volume Twitter texts is proposed. Firstly a bidirectional Long Short Term Memory (Bi-LSTM) is exploited to extract the sentence level and mention level features. Then, to selectively express the essential parts of generated features, a selective gate is applied on sentence level features. Next, to integrate the time information of event mention pairs, an auxiliary feature is designed based on triggers and time attributes of the two event mentions. Finally, all these features are concatenated and fed into a classifier to predict the binary coreference relationship between the event mention pair. They also released a new dataset called EventCoreOnTweet(ECT) dataset on which they evaluated their methods.It annotates the coreferential relationship between event mentions and event trigger of each event mention. The experimental results demonstrate that the approach achieves significant performance in the ECT dataset.

Architecture



Semantic representation for a single event mention on Twitter.



coreferential decision of two event mention

Model

1. **Sentence and Mention Level Features**

   After pre-processing each segmented token in a tweet will be transformed into an embedding vector. The embedding vector consists of two parts: word embedding and distance embedding. Word2vec was used to pre-train word embedding, and label *<unknown>* to represent the out of vocabulary (OOV) words. Words that are close to event trigger have more semantic relevance which is captured by distance embeddings. Bi-LSTMs contains two parts $LSTM_f$ and $LSTM_b$. $LSTM_f$ captures left context and $LSTM_b$ captures right context. Output of both of them are concatenated and fed to selective gate. The sentence level feature (dashed rectangle in the left of figure above) is a sequence of each word's context feature in tweet when tweet as input where n is the length of tweet.

   $$Sent_{level} = (h_0 , h_1 , h_2 , ..., h_n)$$

   And the mention level feature (dashed rectangle in the right of the figure above) is the concatenation of the last output of forward and backward LSTM cell when event trigger as input where m is the length of event trigger.

   $$Ment_{level} = [h_{f,m} , h_{b,0}]$$

2. **Selective Expression**

   Each word plays a different role for one specific event trigger in the same sentence. Some are core words, while another portion of words is semantically confusing. Therefore selective expression mechanisms were employed to achieve more accurate latent features of event mentions by limiting the semantic expression of unimportant or irrelevant words according to the event trigger.

3. **Attention Mechanism**

   If there are different event mentions in a tweet, the semantic contribution of each word in the tweet to an event mention's semantic representation is varied. Therefore attention mechanisms is used to compute the importance score for each word's selective expression to reflect the different contributions to semantic representation of each event mention.

4. **Coreference Decision**

   To determine coreference between two event mentions the second part of the architecture receives the semantic representation of two event mentions V_{em}^{1} and V_{em}^{2} .Along with $V_{local}^{1,2,}$ where $V_{local}^{1,2}$ captures the number of words in the overlap between event triggers and number of days between two event mention, the model does binary classification as coreferential and non-coreferential. Finally a softmax layer is applied to calculate the probability of two categories.

Methodologies Tried

Since the aim is to implement a paper most of the methodologies or the way to get the result has already been tried by the author. Our task is to replicate the results of the author. First we had to decide upon the framework to use. We decided to use Tensorflow since the author also used the same(and it was suggested by our mentor also). Next we created dataset using the method described above under the headings *Data Collection and Dataset creation*. We have discussed the NN implementation details above under the heading *Implementation.*

Dataset used

We are using the dataset provided by the author - EventCoreOnTweet(ECT). It can be found on https://github.com/pinggeger/ECT and dataset after tweet insertion https://github.com/him-mah10/IRE-Major-Project/tree/master/Dataset

Work Done Till Now

1. **Data Collection:**

   To reproduce the results we need to use the dataset used by the authors. However since they did not have permission to distribute the tweets themselves, they removed the terms, replacing them with *"WORD"* and left with instructions to create their dataset. Thus to create their dataset we followed these instructions. First we extracted tweet ids from *corpus.txt* which they uploaded. Script to do so was created on the fly. Then we downloaded tweets corresponding to these tweet ids using api provided by *tweepy*. In total we got around 1994 tweets out of total of 2994 tweets that were mentioned. ***Since the collected data is only 67% of***

the actual dataset we do not expect results to same as the results of the author.
Tweets which were missing were due to one of the following reasons:
1) User no longer exists
2) Tweet no longer exists
3) We didn't have the permission to view the tweet.
After getting tweet we tokenized and pre-processed the tweets using the scripts provided by them and then we wrote a script to insert actual tweets using tweet ids in orignal *corpus.txt* file provided by the author. In case tweet is not available, we left it blank. All the code and dataset created could be found at:
https://github.com/him-mah10/IRE-Major-Project/tree/master/Dataset

2. **Dataset creation:**

   a. From the data obtained using tweepy, we created the dataset for training(sentence pairs and label) and testing using the algorithm mentioned in the paper.
   b. For each event mention e, we paired it with all other event mentions which were tweeted in a time period of less than 7 days after e.
   c. For each pair, we stored the event_mention_id for EM1, tweet_text for EM1, trigger_text for EM1, event_mention_id for EM2, tweet_text for EM2, trigger_text for EM2, label and timestamp difference between the pair.
   d. The event-mentions in a pair are considered co-referential(i.e. label=1), iff they have the same event_instance_id.

3. **Implementation:**

   a. We indexed the unique words in the tweet_text and trigger_text for both train and test data.
   b. We used Glove twitter 100d embeddings to assign embeddings to words. If any word is not present in Glove, we assigned a [0]*100 embedding value to it.
   c. We also calculated relative distance of each word from event trigger and sent it as input to model. The model initializes random distance embeddings and trains them along with others.
   d. We appended the word embedding and distance embedding and created final embedding for each word and passed it as input to BiLSTM.
   e. We are in the process of implementing selective gate mechanism

Findings from current implementation

- Since the collected data is only 67% (1994 out of total of 2995) of the actual dataset we do not expect results to same as the results of the author.
- On training word2vec on the training dataset created we saw that there were about 300 unknown words(out of 900 unique words) in test data. So we decided on using glove embeddings (glove.twitter.27B.100d) which was not used by the author. We believe it will provide better results given that number of tweets are also less than the original number of tweets used by author.
- In the current implementation, we are assigning same embedding to all the <unknown> words. A better way is to assign each <unknown> word the average of embeddings of its surrounding words and make the embeddings trainable in the neural model.
- Since we need GPUs we are using Google colab and ADA.

Code link

https://github.com/him-mah10/IRE-Major-Project

How different is your methodology and scope from the original timeline in the scope document

Our methodology and scope is not significantly different from the scope doc. However, we made the following deviations from the paper mentioned in scope doc:

- On training word2vec on the training dataset created we saw that there were about 300 unknown words(out of 900 unique words) in test data. So we decided on using glove **twitter** embeddings (glove.twitter.27B.100d) which was not used by the author. We believe it will provide better results given that number of tweets are also less than the original number of tweets used by author.
- We merged validation data with train data since there is very less training data and hyperparameters are also fixed.

We aimed at completing understanding the paper, literature survey and 30-40% of the code by the 21st October(First deliverable) and we thus are running on schedule.

Timeline for final deliverable goals

Date	Deliverable
5 Oct	*Understanding the problem statement, reading the paper and literature survey.*
12 Oct	*Getting the dataset using API provided by tweepy . Explored different deep learning frameworks and decided upon tensorflow*
21 Oct	*First deliverable* ***1.*** *Processing data and creating the train, test datasets* ***2.*** *Coded Bi-LSTM model in tensorflow and wrote code for training it.*
7 Nov	Completing the implementation of the paper and be ready with the results.
10 Nov	Doing a literature review and reading of relevant papers and concept for further work.
11 Nov	Final deliverable

We aim at completing the full implementation by 7th November and be ready with the results. After that we aim at doing a literature review for the paper.

Note:

- ***Why did the author use LSTM for mention level features?***
  - The main advantage which the LSTM provides is that it can capture long distance relationships i.e. it has some memory in it. Since LSTM anyway has all the features of RNN and has added benefits of memory, we believe, that the author had used LSTM to capture mention level features.
- **Why Bi-LSTM for mention level features and not LSTM?**
  - A mention might contain more than one words so it makes sense to capture both left and right context.