Natural Language Processing

Professor : Sudeshna Sarkar

Teaching Assistant : Alapan Kuila

Term Project Report

Team Members:

Kattunga Lakshmana Sai Kumar
20MA20026Mohd Arsalan 20MA20034
Nabhya Khoria 20MA20035
Macharla Nitin Teja 20EC10100

1 Problem Statement

EVENT extraction (EE) is an activity that seeks to discover the text's EVENTs and argument information connected to those EVENTs and display them in a structured manner. Most earlier studies attempt to address the issue by individually identifying several substructures and combining them to obtain the entire EVENT structure. The technique's flaw is that they don't fully account for all of the dependencies between the EVENT-triggers, parameters, and roles. Traditional approaches to the task of EVENT extraction primarily rely on elaborately designed features and complicated natural language processing (NLP) tools. In addition, extracting such features depends heavily on the performance of pre-existing systems, which could suffer from error propagation.

2 Dataset

2.1 Description

The given dataset consisted of news articles, reports about various natural calamities, accidents, terror activities etc. Each article is provided to us in the xml format which contained the breakdown into trigger EVENT word, arguments and their roles. In the input sentence, every EVENT is given an unique id to differentiate so that whenever an argument is encountered, it can be linked to its corresponding trigger EVENT using the ID given to the EVENT.

2.2 Extracting Sentence, Tuples, POS Tags, and Dependency Tags

We have written a code which takes these xml files as input and finds the sentence embedded in the file and also the seven element tuples which are the output for the sentence. We have also also added a small piece of code to get the POS tags and Dependency tags for each token in the sentence using spacy library. POS and dependency tags are used in the model so as to gather sentence level information and interdependencies between the tokens. As most of the EVENT and argument phrases are nouns, POS tags help a lot in differentiating those from the rest and hence the corresponding pos and dependency embeddings are concatenated with the word level embeddings from glove. The notebook for the data parsing is provided here click.

2.2.1 Event types

After extracting the sentences,output tuples, POS and dependency tags for each sentence in train, dev and test sets, the distinct set of EVENT types occurred in the dataset are:

- NATURAL EVENT:CYCLONE
- NATURAL EVENT:STORM
- NATURAL EVENT:EPIDEMIC
- MANMADE EVENT:FIRE
- MANMADE EVENT:NORMAL BOMBING
- MANMADE EVENT:SUICIDE ATTACK
- NATURAL EVENT:BLIZZARD
- MANMADE EVENT:TRANSPORT HAZARDS
- NATURAL EVENT:EARTHQUAKE
- MANMADE EVENT:VEHICULAR COLLISION
- MANMADE EVENT:SHOOT OUT
- NATURAL EVENT:FLOODS
- NATURAL EVENT:TORNADO
- NATURAL EVENT:VOLCANO
- NATURAL EVENT:LAND SLIDE
- NATURAL EVENT:AVALANCHES
- NATURAL EVENT:HAIL STORM
- MANMADE EVENT:INDUSTRIAL ACCIDENT
- MANMADE EVENT:AVIATION HAZARD
- NATURAL EVENT:HURRICANE
- MANMADE EVENT:TERRORIST ATTACK
- NATURAL EVENT:COLD WAVE
- NATURAL EVENT:FOREST FIRE
- MANMADE EVENT:SURGICAL STRIKES
- NATURAL EVENT:TSUNAMI
- MANMADE EVENT:TRAIN COLLISION
- MANMADE EVENT:ARMED CONFLICTS
- MANMADE EVENT:RIOTS
- NATURAL EVENT:ROCK FALL
- NATURAL EVENT:HEAT WAVE
- NATURAL EVENT:DROUGHT
- NATURAL EVENT:FAMINE
- MANMADE EVENT:ACCIDENTS
- MANMADE EVENT:MISCELLANEOUS
- MANMADE EVENT:CRIME
- NATURAL EVENT:LIMNIC ERRUPTIONS

2.2.2 Argument types

The distinct set of roles of the arguments linked to the trigger event phrases are obtained as follows:

- TIME
- PLACE
- CASUALATIES
- MAGNITUDE
- AFTER EFFECTS
- NAME
- REASON
- INTENSITY
- PARTICIPANT
- SPEED
- DEPTH
- TYPE
- EPICENTRE
- TEMPERATURE

Since we are using a seven element tuple as output format, the sixth element in each tuple in each tuple which is the argument entity classification is not present in the dataset given to us. So, we are using two broad classes for that which are "ARG" or "NA". "ARG" is to mention that there is an argument linked to the trigger event phrase. "NA" is used in those cases where there is an event in the sentence but there is no argument associated with it.

2.2.3 Result of Dataprocessing

An example of an input sentence which is extracted from an xml file (file86.xml in the given dataset) :

"[unused1] [unused2] Chinese national shot dead in Karachi. Police term it a 'targeted killing.' A Chinese national was shot dead by unidentified assailants here on Monday in an apparent targeted killing, a media report said. The police called it a "targeted killing" as he was shot after being followed by the assailants. A passerby was also injured in the incident in Karachi's Clifton area near Zamzama Park, the Express Tribune reported. Police said nine bullet casings were found at the site of the incident. Inquiry ordered. The place was immediately cordoned off. Minister of Interior Sindh Sohail Anwar Khan ordered the police to start an inquiry into the matter. Security was beefed up for foreigners, particularly Chinese nationals across the province."

- The output tuples obtained from the above sentence are:

  4 5 MANMADE EVENT:SHOOT OUT 2 3 ARG PARTICIPANT — 4 5 MANMADE EVENT:SHOOT OUT 7 7 ARG PLACE — 18 19 MANMADE EVENT:SHOOT OUT 15 16 ARG PARTICIPANT — 18 19 MANMADE EVENT:SHOOT OUT 25 25 ARG TIME — 59 59 MANMADE EVENT:SHOOT OUT 53 56 ARG CASUALTIES — 59 59 MANMADE EVENT:SHOOT OUT 61 66 ARG PLACE

- The POS tags obtained for the sentence are:

  $< UNK >< UNK >$ ADJ ADJ NOUN ADJ ADP PROPN PUNCT NOUN NOUN PRON DET PUNCT VERB NOUN PUNCT PUNCT DET ADJ NOUN AUX VERB ADJ ADP ADJ NOUN ADV ADP PROPN ADP DET ADJ VERB NOUN PUNCT DET NOUN NOUN VERB PUNCT DET NOUN VERB PRON DET PUNCT VERB NOUN PUNCT SCONJ PRON AUX VERB ADP AUX VERB ADP DET NOUN PUNCT DET NOUN AUX ADV VERB ADP DET NOUN ADP PROPN PART PROPN NOUN ADP PROPN PROPN PUNCT DET PROPN PROPN VERB PUNCT NOUN VERB NUM NOUN NOUN AUX VERB ADP DET NOUN ADP DET NOUN PUNCT NOUN VERB PUNCT DET NOUN AUX ADV VERB ADP PUNCT PROPN ADP PROPN PROPN PROPN PROPN PROPN VERB DET NOUN PART VERB DET NOUN ADP DET NOUN PUNCT NOUN AUX VERB ADP ADP NOUN PUNCT ADV ADJ NOUN ADP DET NOUN PUNCT

- The dependency tags for the same are obtained as:

  $< UNK >< UNK >$ amod amod ROOT amod prep pobj punct compound ROOT dobj det punct amod oprd punct punct det amod nsubjpass auxpass ccomp oprd agent amod pobj advmod prep pobj prep det amod amod pobj punct det compound nsubj ROOT punct det nsubj ROOT dobj det punct amod oprd punct mark nsubjpass auxpass advcl prep auxpass pcomp agent det pobj punct det nsubjpass auxpass advmod ccomp prep det pobj prep poss case compound pobj prep compound pobj punct det compound nsubj ROOT punct nsubj ROOT nummod compound

nsubjpass auxpass ccomp prep det pobj prep det pobj punct nsubj ROOT punct det nsubjpass auxpass advmod ROOT prt punct nmod prep pobj compound compound compound nsubj ROOT det dobj aux xcomp det dobj prep det pobj punct nsubjpass auxpass ROOT prt prep pobj punct advmod amod appos prep det pobj punct

2.3 Bert Tokeniser

The dataset shown above is the raw data extracted from the xml files. Before, feeding it into the model, we have bert tokenized each and every sentence and modified the corresponding pointers (output tuples), POS tags and dependency tags accordingly. The code used for the bert tokenising task is Notebook.

- Input sentence

  [unused1] [unused2] Chinese national shot dead in Karachi ##. Police term it a ‘ ##tar ##get ##ed killing ##. ##’ A Chinese national was shot dead by unidentified ass ##ail ##ants here on Monday in an apparent targeted killing ##, a media report said ##. The police called it a “ ##tar ##get ##ed killing ##” as he was shot after being followed by the ass ##ail ##ants ##. A pass ##er ##by was also injured in the incident in Karachi ##’ ##s Clifton area near Z ##am ##zam ##a Park ##, the Express Tribune reported ##. Police said nine bullet ca ##sing ##s were found at the site of the incident ##. Inquiry ordered ##. The place was immediately cord ##oned off ##. Minister of Interior Sindh So ##hai ##l An ##war Khan ordered the police to start an inquiry into the matter ##. Security was beef ##ed up for foreigners ##, particularly Chinese nationals across the province ##.

  [“ ## ” are added in bert tokeniser to indicate that the word/token comprises of multiple morphemes and it breaks each token into individual morphemes separated by ##.]

- Output Tuples

  4 5 MANMADE EVENT:SHOOT OUT 2 3 ARG PARTICIPANT — 4 5 MANMADE EVENT:SHOOT OUT 7 8 ARG PLACE — 24 25 MANMADE EVENT:SHOOT OUT 21 22 ARG PARTICIPANT — 24 25 MANMADE EVENT:SHOOT OUT 33 33 ARG TIME — 78 78 MANMADE EVENT:SHOOT OUT 70 75 ARG CASUALTIES — 78 78 MANMADE EVENT:SHOOT OUT 80 91 ARG PLACE

- POS Tags

  $< UNK >< UNK >$ ADJ ADJ NOUN ADJ ADP PROPN PROPN PUNCT NOUN NOUN PRON DET DET DET DET PUNCT PUNCT PUNCT VERB NOUN PUNCT PUNCT DET ADJ NOUN AUX VERB VERB VERB ADJ ADP ADJ NOUN ADV ADP PROPN ADP ADP DET ADJ VERB NOUN NOUN PUNCT DET NOUN NOUN VERB PUNCT PUNCT PUNCT PUNCT DET DET NOUN VERB PRON DET PUNCT VERB NOUN PUNCT SCONJ PRON PRON PRON PRON AUX VERB VERB VERB ADP AUX VERB ADP DET NOUN PUNCT DET DET DET NOUN AUX ADV VERB VERB VERB VERB ADP ADP DET NOUN ADP PROPN PROPN PART PROPN NOUN ADP PROPN PROPN PROPN PROPN PUNCT DET PROPN PROPN VERB PUNCT NOUN NOUN VERB NUM NUM NOUN NOUN AUX VERB ADP ADP DET DET NOUN ADP DET NOUN PUNCT PUNCT PUNCT NOUN NOUN VERB PUNCT DET NOUN AUX ADV VERB ADP PUNCT PROPN ADP ADP PROPN PROPN PROPN PROPN PROPN PROPN VERB VERB DET NOUN PART VERB DET NOUN NOUN

- Dependency tags

  $< UNK >< UNK >$ amod amod ROOT amod prep pobj pobj punct compound ROOT dobj det det det det punct punct punct amod oprd punct punct det amod nsubjpass auxpass ccomp ccomp ccomp oprd agent amod pobj advmod prep pobj prep prep det amod amod pobj pobj punct det compound nsubj ROOT punct punct punct punct det det nsubj ROOT dobj det punct amod oprd punct mark nsubjpass nsubjpass nsubjpass nsubjpass auxpass advcl advcl advcl prep

auxpass pcomp agent det pobj punct det det det nsubjpass auxpass advmod ccomp ccomp ccomp ccomp prep prep det pobj prep poss poss case compound pobj prep compound compound compound compound pobj punct det compound nsubj ROOT punct nsubj nsubj ROOT nummod nummod compound nsubjpass auxpass ccomp prep prep det det pobj prep det pobj punct punct punct nsubj nsubj ROOT punct det nsubjpass auxpass advmod ROOT prt punct nmod prep prep pobj compound compound compound compound nsubj ROOT ROOT det dobj aux xcomp det dobj dobj

2.4 Dataset Statistics

The code has been used to analyse the statistics of dataset after processing it through bert tokenizer model is Notebook.

We have obtained the following results after running the above code:

Statistics	
Number of Sentences in Train Set	761
Number of Sentences in Test Set	205
Number of Sentences in Dev Set	182
Average Sentence Length Train Set	294
Maximum Sentence Length Train Set	1290
Average Sentence Length Test Set	324
Maximum Sentence Length Test Set	1071
Average Sentence Length Dev Set	208
Maximum Sentence Length Dev Set	602
Number of Event Types in Train Set	35
Number of Event Types in Test Set	29
Number of Event Types in Dev Set	27

Event Type	Train	Test	Dev
NATURAL EVENT:CYCLONE	15	4	8
NATURAL EVENT:STORM	21	8	16
NATURAL EVENT:EPIDEMIC	2	0	1
MANMADE EVENT:FIRE	90	41	19
MANMADE EVENT:NORMAL BOMBING	22	12	6
MANMADE EVENT:SUICIDE ATTACK	14	4	2
NATURAL EVENT:BLIZZARD	2	0	1
MANMADE EVENT:TRANSPORT HAZARDS	69	14	13
NATURAL EVENT:EARTHQUAKE	49	8	23
MANMADE EVENT:VEHICULAR COLLISION	136	24	16
MANMADE EVENT:SHOOT OUT	45	18	22
NATURAL EVENT:FLOODS	31	6	12
NATURAL EVENT:TORNADO	11	3	10
NATURAL EVENT:VOLCANO	12	2	12
NATURAL EVENT:LAND SLIDE	14	5	4
NATURAL EVENT:AVALANCHES	2	2	2
NATURAL EVENT:HAIL STORMS	4	1	3
MANMADE EVENT:INDUSTRIAL ACCIDENT	15	8	3
MANMADE EVENT:AVIATION HAZARD	13	5	3
NATURAL EVENT:HURRICANE	4	1	5
MANMADE EVENT:TERRORIST ATTACK	13	3	8
NATURAL EVENT:COLD WAVE	7	0	4
NATURAL EVENT:FOREST FIRE	4	3	3
MANMADE EVENT:SURGICAL STRIKES	17	7	4
NATURAL EVENT:TSUNAMI	2	0	0
MANMADE EVENT:TRAIN COLLISION	14	2	4
MANMADE EVENT:ARMED CONFLICTS	1	3	1
MANMADE EVENT:RIOTS	3	1	1
NATURAL EVENT:ROCK FALL	1	0	0
NATURAL EVENT:HEAT WAVE	3	2	0
NATURAL EVENT:DROUGHT	1	1	0
NATURAL EVENT:FAMINE	1	0	0
MANMADE EVENT:ACCIDENTS	192	30	0
MANMADE EVENT:MISCELLANEOUS	27	7	0
MANMADE EVENT:CRIME	20	13	0
NATURAL EVENT:LIMNIC ERRUPTIONS	0	0	0

Since BERT model can take a maximum sentence length of 512(count of tokens in the sentence), We have written a code to truncate every sentence of train, dev and test sets so that it can be fed into the model. We have considered the maximum sentence length to be 200 (to train the model faster because of lack of GPU). Whenever a sentence is truncated to a maximum length of 200, we need to also take care of the pointer(tuples) and make sure that start and end index of the event or argument should be within the range. The code is built to take care of that, it also modifies the pointers of a sentence according to its truncation.

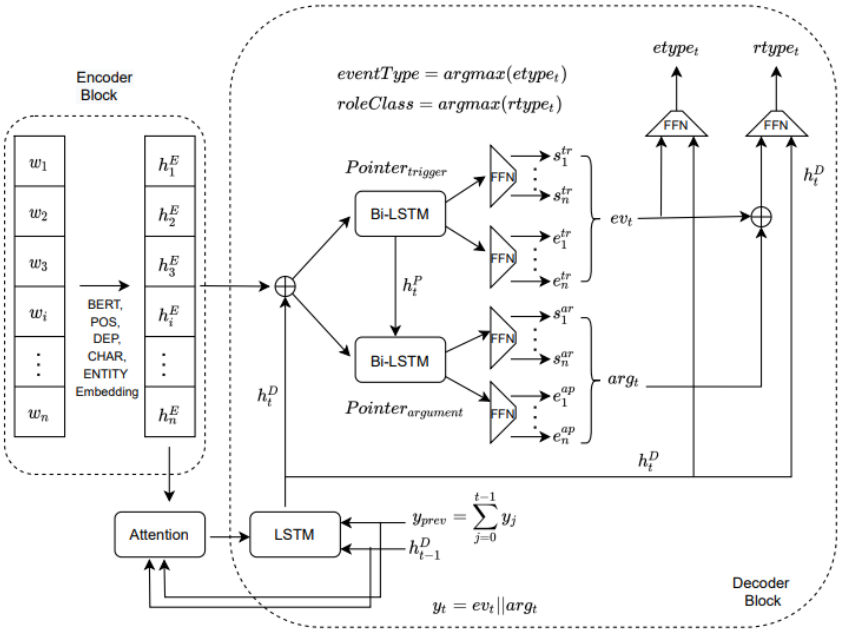
The Bert Dataset Truncation code is given below:
https://colab.research.google.com/drive/1bDPetgTavGXZc5ysSh5GTy-BevWky-vh?usp=sharing
Now, These sentences are fed into the model for training and testing.

[Note: In the zip file that we submitted, there is a folder named dataset which contains of few subsolders as follows:
XML files: This folder has the given dataset which are .xml files.
Raw data: This folder contains the sentences and pointers just after preprocessing the xml files.
Bert data: This folder contains the sentences and updated pointers after feeding raw data files to bert Tokenizer model discussed above.
Model data: This folder contains the truncated sentence files and accordingly modified pointer files so that it can be directly fed to the model. (This folder is user specific, since I have taken maximum sentence length to be 200, We are using the files in these, if anyone wants to change the hyperparamter of maximum sentence length, he/she can run the above given Bert Dataset Truncation code to generate files of desired maximum sentence length and use those to feed the model.)]

3 The Model

3.1 Methodology

PESE framework employ a encoder-decoder architecture for the end-to-end EE task. The overview of the model architecture is depicted in figure below. The input to our model is a sentence (i.e. a sequence of tokens) and as output,is a list of event tuples present in that sentence. A pre-trained BERT is used at the encoder and LSTM-based network at the decoder in the model.



3.1.1 Sentence Encoding

We use pre-trained BERT model as the sentence encoder to obtain the contextual representation of the tokens. We combine the POS, DEP, and character-level features with the BERT embeddings to represent each token in the input sentence. So along with pre-trained BERT embedding we use four other embeddings: 1) POS embeddings 2) DEP embeddings 3) character-level embeddings.

3.1.2 Extraction of Event Frames

Our proposed decoder generates a sequence of event tuples. The decoder comprises sequence-generator LSTM, two pointer networks, and two classification networks. The event frame sequence is generated by the sequence-generator LSTM. The trigger and argument spans of the events are identified by the pointer networks. The classification networks determine the type of event and the trigger-argument role label.

3.1.3 Training Procedure

To train our model, we minimize the sum of negative log-likelihood loss for identifying the four position-indexes of the corresponding trigger and argument spans and two classification tasks: 1)event type classification and 2) role classification.

3.2 Improvements

DMCNN was the previous state-of-the-art method that was widely used for EE. DMCNN - dynamic multi-pooling convolutional neural network (DMCNN) proposes the task in two stages. The first stage is called trigger classification, in which DMCNN is used to classify each word in a sentence to identify trigger words. If one sentence has triggers, the second stage is conducted, which applies a similar DMCNN to assign arguments to triggers and align the roles of the arguments. This model follows a pipelined approach where triggers and corresponding arguments are identified in separate stages. However, this pipeline approach has to deal with error propagation problems. PESE framework however utilizes the event-argument-role interdependencies to extract the event frames from text. The encoder encodes the input sentence, whereas the decoder identifies an event frame in each time step based on the input sentence encoding and the event frames generated in the previous time steps. PESE framework represents each event-frame using a 7-tuple structure that stores all the records, as mentioned earlier. The 7-tuple contains: 1) start index of trigger phrase, 2) end index of trigger phrase, 3) event type, 4) start index of argument phrase, 5) end index of argument phrase 6) argument classification, 7) trigger-argument role label.

3.3 Training

3.3.1 Observations while Training

The following results were observed while training the model:

- Training data size: 761
- Development data size: 182
- vocab length: 9409
- embed dictionary length: 4465

3.3.2 Hyperparameters

Hyperparameters used in the model are:

- Max sentence length = 200
- Batch size = 8 (The GPU available with us was crashing for higher batch size)
- Number of epochs = 15 (We didnt have more GPU to train for more number of epochs)
- Word embedding dim = 300 (from glove vectors)
- Character embedding dimension = 30

- POS embedding dimension = 30
- Dependency embedding dimension = 30
- Maximum word length = 10
- Convolution filter size = 3
- Drop out rate = 0.5
- Word minimum frequency to be considered into voca
- Early stop count = 7
- Optimiser = Adam
- Learning rate = 0.001

4 Results

4.1 Results on Train and dev set

The following results were observed while training Training data size: 761
Development data size: 182
vocab length: 9409
embed dictionary length: 4465

Training loss, dev set F1score during training are obtained as follows:

Epoch	Training loss	Dev Results F1
1	15.7486543655395	0.03405107221669818
2	12.097764838369269	0.08216782732146542
3	11.008498834308826	0.10837039536111581
4	10.305366837350945	0.15495342439231183
5	9.787271574923867	0.18498985305232787
6	9.362766577068127	0.18834080222002608
7	9.05609467155055	0.22787979473949013
8	8.710852241516113	0.1973018500629
9	8.388215220601936	0.2181508259528
10	8.17795151158383	0.2475328897910
11	7.964403674477025	0.2955382676643
12	7.579523402766178	0.3112416058180
13	7.523083977950247	0.2811115536309
14	7.3467427554883455	0.314308676702
15	7.099076971254851	0.31565761511497603

The results obtained during training are stored in the below log file:
$https://drive.google.com/file/d/1EbMUFPW9nTkq29Ypf4Ah923bS0UUwbjm/view?usp=sharing$
The vocabulary of the training dataset is stored in the below file:
$https://drive.google.com/file/d/1V_Qjr13JjjuEtI45WV1y-jGMWx_XaWEz/view?usp=sharing$
The trained model (named as model_bert_lstm.h5py) is available below:
$https://drive.google.com/file/d/1-0DZ4EBGctBHHpz_kSt9l9Ayx4VjEgs7/view?usp=sharing$
The link for the training notebook of the model is given below:
$https://colab.research.google.com/drive/1-Ns0Mifo7qBiTdAVrrhKPdmnGxK_gMWT?usp=sharing$

4.2 Results on Test set

The results on the test set are obtained as follows:
Test size: 205
no of correctly identified triggers= 1005
no of correctly classified triggers= 864
no of correctly identified arguments= 457
no of correctly identified roles= 383
P tuple: 0.298
P ti: 0.977
P tc: 0.84
P ai: 0.444
P ro: 0.372
R tuple: 0.23
R ti: 0.753
R tc: 0.647
R ai: 0.342
R ro: 0.287
F1: 0.26
TI F1: 0.85
TC F1: 0.731
AI F1: 0.387
RL F1: 0.324

Here P means Precision, R means recall
Whereas, ti = trigger event index , tc = trigger event classification, ai = argument index ro = role of the argument phrase

These results are stored in the following log file: https://drive.google.com/file/d/1-1OJxLx1Rfco_OuneNJlC9Uvc34r2Ne0/view?usp=sharing

The predicted outputs (tuples) on the test set are stored in the below file: https://drive.google.com/file/d/1QvDHMcEnuftcYQTV6DGLEhbahA9yNlxC/view?usp=sharing

4.3 Observations

The test results obtained can be further improved if the number of epochs for which the model has been trained on the training set is increased. Since we had limited GPU resources, we had to train the model for a less number of epochs which is 15. If it can be further trained, there is a good possibility that the results will improve. Also, the hyperparameters can be further tuned such that there is an improved in the results obtained.