

A neural network based Event extraction system for Indian languages

Alapan Kuila, Sarath chandra Bussa, and Sudeshna Sarkar

Indian institute of Technology kharagpur, India

alapan.cse@iitkgp.ac.in, bussasarath2@gmail.com, sudeshna@cseiitkgp.ac.in

Abstract. In this paper we have described a neural network based approach for Event extraction(EE) task which aims to discover different types of events along with the event arguments form the text documents written in Indian languages like **Hindi**, **Tamil** and **English** as part of our participation in the task on Event Extraction from Newswires and Social Media Text in Indian Languages at Forum for Information Retrieval Evaluation (FIRE) in 2018. A neural netork model which is a combination of Convolution neural network(CNN) and Recurrent neural network(RNN) is employed for the Event identification task. In addition to event detection, the system also extracts the event arguments which contain the information related to the events(i.e. when[Time], where[Place], Reason, Casualty, After-effect etc.). Our proposed Event Extraction model achieves f-score of **39.71**, **37.42** and **39.91** on Hindi, Tamil and English dataset respectively which shows the overall performance of Event identification and argument extraction task in these three language domain.

Keywords: Event extraction · Convolution neural network(CNN) · Recurrent neural network(RNN).

1 Introduction

A huge number of news on events, occuring in different corners of the world, are reported each moment in online and printed media. To keep track of those news, it is very important to identify relevant events and extract the spatio-temporal aspects about those events. Understanding events and their descriptions in raw text is the key factor in automatic event extraction which is an important and challenging task in Natural Language Processing(NLP) and Information Extraction(IE). It is also essential in practical applications like news summerization, information retrieval and knowledge base construction. Event extraction is an important and challenging task in Information extraction(IE) which aims to detect, from the text, the occurrence of events of specific types, and to discover the arguments(event participants or attributes) that are associated with the event. Event Arguments are basically represents the event related information i.e. capturing who does what to whom, how, when and where. For example,

- **S1:** *Mild earthquake has been found in Indonesia's Sulawesi island.*

A typical automatic Event extraction system will identify from **S1** that the sentence depicts a **Earthquake** event and it also extract some important information related to the earthquake such as, **Place of the event:** *Indonesia's Sulawesi island.*

Current event extraction systems rely on examining *event trigger* words to check if an event occurs or not and if occurs then what is the type of the event. An *event trigger* is a word or phrase that clearly express the occurrence of an event. For example, in **S1**, *earthquake* is the trigger word as it is the most salient clue in the sentence to identify the **Earthquake**. The Understanding and identifying event triggers in raw text is an important and challenging problem as the same event may be expressed by various trigger expressions and an specific event trigger might be used to represent different events in different context. From the previous research it is evident that Event detection task is not a simple trigger-token searching task. And Contextual feature information is very much important in identification and classification of event triggers. Event argument extraction is another challenging task. Unlike entity identification task, event argument extraction task deals with discovering the specific entities which are relevant to the events appeared in the document. So typical NER systems are not sufficient to solve the problem. Previously, we have worked on detecting the segment of the text which contains some event related information from twitter[18]. This task is the extension of the previous work. Here we have tried to identify the events as well as event arguments from news documents using supervised learning procedure with the help of annotated corpus where event triggers and argument phrases are tagged. So our contribution in this task is two fold. First, we have designed a hybrid neural network based model for event extraction from news document written in Indian languages like Hindi and Tamil as well as English and second, we have represented the extracted events in structured format (Table1).

2 Related Studies

A plethora of works have already been done in event extraction from text documents. However supervised methods are dominant in this problem which requires annotated training data to employ machine learning based approaches. Previously, the researchers have designed various feature (lexical, contextual, syntactic) based approach for Event Detection[3, 4]. But the problem with these approaches is that they require thorough feature engineering, and extensive language knowledge which also impede the designing of language independent ED model.

Now neural networks are widely applied in NLP tasks including Event Extraction(EE) as they automatically extract underlying features. Some of the papers apply Bidirectional recurrent neural networks(Bi-RNN) as it is capable to capture both preceding and following context information of each token in the sentence[1, 7]. Some have also used convolution neural network(CNN) as CNN is good at capturing features from a sequence of objects [1, 2, 5, 7]. Liu et al., 2017

has used event argument information to enhance the quality of event detection accuracy[6]. Peng et al., 2016 has tried to detect events from text documents using minimal supervision approach where system needs only annotation guidelines[8]. It formalizes the problem as a semantic similarity problem[7]. Some works also exists where researchers have tried to solve it through frame semantic parser[9].

Here, we are interested in news events depicting man-made or natural disasters. In the disaster domain, maximum research works are centered around extracting events from user generated content such as social media texts. Extracting event from social media text consists of two subtask: Identification of informative tweets and detecting events from those informative tweets[10]. Artificial Intelligence for Disaster Response (AIDR) [11] is one such platform that has been successfully deployed that works on streaming microblog text (twitter) to classify messages that people post during disasters into a set of user-defined categories of information. Muhammad et al. (2013) has extracted disaster relevant information from twitter and classified them into personal, informative and others[12]. Something similar has been done by Chowdhury et al. (2013), who has classified crisis related tweets and categorize them into: Pre-incident, during-incident and post-incident events[13].

Event extraction are also represented as a natural language understanding problem. Previously, natural language understanding task was highly dependent on script knowledge[14]. Though script knowledge is very useful in event-event relation extraction and causality detection but scripts are heavily domain dependent and too expensive to create[15]. To get the event structure knowledge from documents, narrative event chain is introduced by Chambers and Jurafsky(2009)[14, 16]. Event Detection is also formalized as Semantic Role labeling(SRL) problem[17]. But the problem is that these works take only verbs as events and from our experience we know that events could be noun, verb or adjective. Chambers and Jurafsky(2009) have introduced the concept of the protagonist[14] for event representation, which is also not feasible in case of disaster related news documents. Though there exists several successful works for English language such as ACE, TAC evaluation tracks ¹ but there is no such standard event extraction tool for Indian Languages. So we see that none of the previous work is adequate to tackle the problem of event extraction from news documents written in Indian languages.

3 Problem Description

Here, we focus on the sentence level Event extraction task. Given a news document we want to extract the events inside that document and represent the events in structured format. Our work covers the events which depicts the occurrence of any disasters caused by nature (e.g. Flood, Earthquake etc.) or manmade (e.g. terrorist attack, accident, riot etc.). The term **Event** covers the type of the

¹ <https://tac.nist.gov/2017/KBP/Event/index.html>

event and the event participants or arguments which depicts the informations regarding that event(i.e. when[Time], where[Place], cause, effect etc). So while extracting events first we have to identify the type of the extracted event based on the trigger word found in the text. Event argument identification is the another crucial module of our system as arguments are really important to describe the events. Each event has some general attributes like *place(where)*, *time(when)* and some specific attributes like *cause*, *after-effect*, *magnitude*. Our interest is to identify these attributes for each events which are present in the document. And if the same event has more than one argument of same type then we have to aggregate the argument information accordingly such that no superfluous information exist in final outcome. Table 1 depicts sentence level event extraction output where each row indicates one event mention with corresponding argument information.

Table 1. Sentence level Event Extraction output

S-id	Event type	Time	Place	Casualty	Participant
1	Shoot-Out	Thursday	neighborhood of Dast-e-Barchi	56 people were wounded	-
2	Shoot-Out	Friday	Eastern city, Ghazni	35 civilians killed	Taliban
3	Suicide-attack	Wednesday	Afghanistan	at least 35 soldiers killed	-

4 Datasets

This section describes the dataset that have been used in this work. The training and the test data have been made available to the task participants by the organizers. The training dataset consists of annotated documents where event trigger words as well as event arguments are tagged. The testset contains news documents without any annotations. The dataset statistics is represented in Table 2.

Table 2. Dataset description

Language	Number of Training Docs	Number of Test Docs.	Number of Event types	Number of Argument types
Hindi	107	311	14	7
Tamil	64	1438	23	10
English	100	803	16	10

5 Methodology

An event extraction problem can be further subdevided into two subtasks: (1)*Event Detection*, the task of finding the event mentions of predefined event types and (2)*Event argument extraction*, the identification of event related information like time, place, cause, effect which are relevant to the event mentions. We formalize the Event Detection(ED) problem as a multi-class classification problem via combination of convolution neural network(CNN) and bidirectional Long short term memory (Bi-LSTM). We expect that the Bi-LSTM and CNN would capture syntactic and semantic information from the text so their combination would help to enhance the overall ED performance. Let, we have k numbers of predefined event types E_1, E_2, \dots, E_k . Now, given a sentence $S = w_1w_2w_3\dots w_n$, where n is the sentence length, for each word w_i in the sentence, we want to predict whether the current token is an valid event trigger. And if it is an event trigger then with which event type E_j the word w_i will be matched. So, the current word along with its sentential context constitute an event trigger candidate which will be input to our classification model. We have fixed the context window size in order to feed the trigger candidate to the CNN.

If the window size is k then trigger candidate for word W would be represented as $[w_{-k}, w_{-k+1}, \dots, w_0, \dots, w_{k-1}, w_k]$ where current word is positioned in the middle position (i.e. w_0). Now these $(2k+1)$ size trigger candidates are taken as input to the CNN and Bi-LSTM models.

Before entering the CNNs and Bi-LSTMs, each word is converted into a real valued vector by looking up a embedding table with an intution that these real valued vectors will capture various semantic characteristics of the words. So the input to our model is a matrix X of size $(2k+1)*|V|$ where $|V|$ is the dimension of the real valued vectors and $X = [x_{-k}, x_{-k+1}, \dots, x_0, \dots, x_{k-1}, x_k]$ where x_j is the vector representation of word w_j . Now the two dimensional representation of each word is feed to a convolution layer followed by a max-pooling layer.

In convolution layer we have used a set of filters F_1, F_2, \dots, F_p and each filter F_i has a window size m_i and can be represented as a matrix of size $m_i * |V|$. After employing max-pooling on the convolution layer output we get a hidden vector representation of size $|F_1| + |F_2| + \dots + |F_p|$ where, $|F_i|$ represents the number of instances of filter type F_i . The hidden vector representation which we get from CNN model is named as F_{CNN} .

Besides CNN, we have also used Bi-directional Long Short Term Memory(Bi-LSTM) in this work. The matrix representation of each word W which is represented as $X = [x_{-k}, x_{-k+1}, \dots, x_0, \dots, x_{k-1}, x_k]$, is also feed to the Bi-LSTM model. The hidden states of Bi-LSTM are computed both in forward and backward ways at each time step. We denote the outputs of the forward and backward LSTM as \vec{h}_t and \overleftarrow{h}_t respectively. At each time step t , hidden vector h_t is computed based on current input vector x_t and previous hidden vector h_{t-1} .

$$\vec{h}_t = \overrightarrow{LSTM}(\vec{h}_{t-1}, w_t) \quad (1)$$

$$\overleftarrow{h}_t = \overleftarrow{LSTM}(\overleftarrow{h}_{t-1}, w_t) \quad (2)$$

Then the output at time t is $h_t = [\vec{h}_t, \overleftarrow{h}_t]$. Here we have taken the hidden representation of x_0 i.e. concatenation of hidden vector representation of forward LSTM \vec{h}_0 And hidden vector representation of \overleftarrow{h}_0 as the output of the Bi-LSTM. Formally, $F_{Bi-LSTM} = [\vec{h}_0, \overleftarrow{h}_0]$.

Now to combine the output of CNN and Bi-LSTM we have concatenated the representation vector F_{CNN} and $F_{Bi-LSTM}$ and feed the concatenated vector to a fully connected layer followed by a softmax layer to get the proper event type of the current word W (see Fig. 1). The gradients are calculated using back-propagation. We have also implemented regularization by dropout.

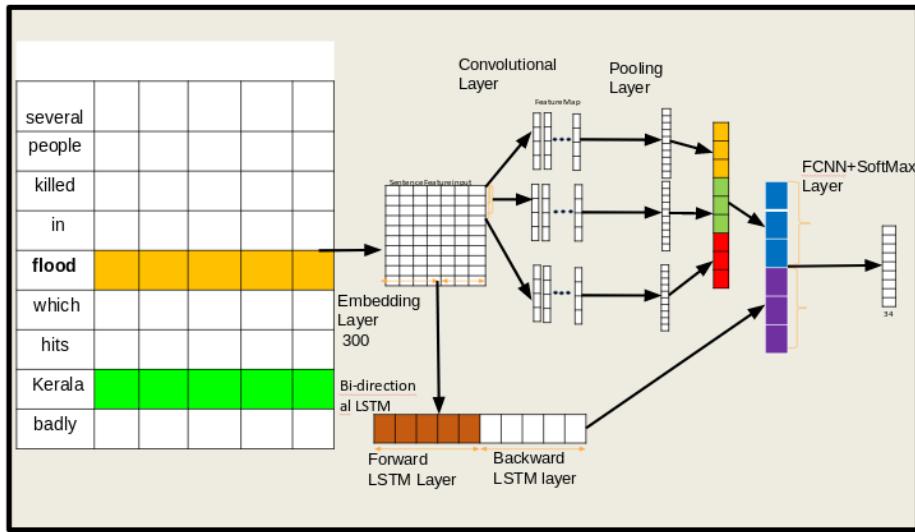


Fig. 1. Deep neural architecture for Event Trigger classification

For Event-Argument Extraction we have applied the same model which is used for Event Trigger classification. Only the number of output classes is different in these two models. We have used BIO annotation scheme for argument identification as most of the arguments contain more than a single token.

Now after **Event-Trigger** and **Event-Arguments** have been identified, our next task is to link the event arguments (which are identified by the Event-Arument extractor) to their corresponding event trigger words so that we can identify the event participants for a specific event trigger. For **Event Trigger-Argument linking** we have taken a simple heuristic based approach where, For each argument phrase we have identified the nearest event trigger word and link those argument-trigger pair. The distance between trigger and argument is calculated by the number of sentences between them and if a tie is there then decision has been taken by number of tokens in between the pairs. Sometimes, a sentence contains useful argument information regarding an event but there

exist no event-trigger word in that very sentence. In that situation the argument is mapped to nearest event-trigger present in the previous or next sentence.

6 Resources and Hyperparameters

We have used the same parameters for the neural network models used in trigger identification and argument extraction. In Convolution neural network we have used filter size of 2, 3, 4 and for each size we have taken 200 filters to generate feature maps from the convolution operations. The final output of the CNN is represented as F_{CNN} which has size 600. The number of hidden units used in Bi-LSTM is 200 which leads to the size of final outcome of the Bi-LSTM layer $F_{Bi-LSTM}$ is 400. Regarding embedding, we have used pre-trained word embeddings of size 300 which we have taken from the fastText toolkit². Finally, we have trained the neural network models using adam optimizer with shuffled minibatches, dropout rate=0.5, backpropagation for gradient calculation and parameter modification.

7 Result Analysis

According to the FIRE 2018 official evaluation methodology the performance of the system is measured by the Precision and Recall and F-score. For example, let there is an Event-mention E_1 and there are six fields such as *Event Type*, *Location*, *Time*, *Event-Participants*, *Causes*, *Effects* for that event. Now for that event E_1 , if all these fields are identified correctly then the system gets full score of 1 else according to the identified fields the score will be calculated. And finally macro-average of the Precision and Recall is calculated for all events in all documents. And the performance of our system is depicted in 3. From the

Table 3. Event extraction performance on the FIRE 2018 official evaluation

Language	Precision	Recall	F-measure
Hindi	62.85	29.02	39.71
Tamil	59.98	27.20	37.42
English	65.16	28.77	39.91

result, we can conclude that we have to improve the Recall score to enhance the system performance. In all the three languages the precision score is quite acceptable but due to low Recall, overall performance of the system is not so satisfactory.

² <https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md>

8 Error Analysis

Our event extraction system has three steps. Firstly we have identified the event trigger words. Then, we have extracted argument phrases. Lastly we have linked the arguments with their nearest trigger words. We have split the training data in the ratio of 80:20 and use the first partition(80%) for training and remaining 20% as development set. In this section we will try to analyze the system's performance on development set and examine the shortcomings of the system.

8.1 Event Identificaton:

While identifying the trigger words, system sometimes fail to get the contextual information which has an important role in event identification. For example:

- **S2:** *Nepalese Maoists will not attack on ordinary people and politicians any more.*

The system tagged *attack* as an event trigger of type *Terrorist-Attack*. But from the context it is clear that this sentence does not describe any terrorist attack event.

- **S3:** *It is reported that 56 people were also wounded in the bombing in the neighborhood of Dasht-e-Barchi.*

The event identification module sometimes failed to correctly classify the event types present in the documents. We have noticed that the system is confused among related event types like *terrorist-attack*, *Shoot-out*, *Suicide-attack* or *Vehicular-Collision*, *Transport-hazard and Accident*. For example, in **S3**, system classified trigger word *bombing* as *Shoot-out* type event but it is actually an event of type *Suicide-attack*.

- **S4:** *Officials blame the Islamic State group for this attack.*

In some of the cases system failed to detect the correct event triggers from the sentences. For example, in **S4**, system failed to identify *attack* as an event trigger word.

8.2 Event-Argument Identificaton:

We have checked the argument extraction output and identified some errors, which we will dicuss below.

Symanitic Variety: Based on development set, have noticed that our system performance is not so satisfactory for some frequent arguments like *Place*, *Time*. The possible cause is the symanitic variety. Some arguments are represented in a very specific way. For example,

- **S5:** *The magnitude of the first quake was 4.9.*

- **S6:** *After a short span there is an another earthquake whose magnitude is 4.1.*
- **S7:** *The Governor of California was injured in a motorcycle accident near the house on Sunday afternoon.*

The *magnitude* of quake is represented very specific way (example S5, S6) and it is always a numeric value. But arguments like place is not all time a place entity which can be detected by typical Named entity recognizer(NER) systems. For example in **S7**, *near the house* is not a typical location entity, but is the place of event type *VEHICULAR COLLISION*.

Token missing in phrase: Most of the event arguments are more than one word phrase. But our system is failed to identify all the words in the phrase which leads to missing important information regarding the event. For example, in the **S7** our system has extracted *Sunday* as time argument whereas the accurate time information will be *Sunday afternoon*.

Reason and After-effect: Our system has failed to identify arguments like *Reason* and *After-effect* for any event instance. Most of the time reason and effect of any event is an event itself and are represented by a long sequence of words or phrase. But as there is no such specific pattern in these two arguments the system is failed to identify them. The cases where there is a semantic clue present in the sentence then system can identify it. For example:

- **S8:** *US Airways has been forced to land in Tehran due to mechanical disruption on its way to Amsterdam with 255 passengers from Mumbai.*

The system has accurately identified the reason of the *AVIATION HAZARD* type event as: *due to mechanical disruption*. The presence of the term *due to* may help in this case. But in most of the cases this type of clues is not present. And the proposed system accuracy is miserably bad for these two attributes. Some smarter technique have to be employed to improve our performance.

8.3 Event trigger-argument linking:

In this module, we have linked the event trigger words with its corresponding arguments using a heuristic approach stated in Section 5.

Incorrect assignment It is noticed that the heuristic approach which takes nearest token for linking, may causes wrong output. Some times due to complex syntactic structure of the sentence a valid argument may appear far away from the trigger word and lots of other token are exist between that argument and the valid trigger word. But our heuristic function has selected an incorrect trigger-argument pair. For example,

- **S9:** *Translation: Earthquake in Indonesia.*

- **S10:** *Mild earthquake has been found in Indonesia’s Sulawesi island.*
- **S11:** *Meanwhile, one Indian has died from the four floors of Manama Hotel in Dubai.*

In the sentence **S11**, there is no trigger word of our interest but our Event-Argument extractor has identified *One Indian has died* as a casualty argument. Our linking module will wrongly map this with *EARTHQUAKE* event in the previous sentence(**S10**), though they are totally unrelated.

Mutual exclusiveness It is highly probable that one single argument may be attribute of more than one different event instance. But according to our heuristic function one argument can only be linked to a specific event trigger word. These problems lead to missing important information in the extracted events and directly affect system performance.

9 Conclusion and Future Work

The main problem that impedes the designing of Event extraction model for Indian language news document texts is scarcity of annotated corpus. The second hindrance is unavailability of syntactic and semantic feature extractor for Indian languages. In this task we have relied on automatically learning effective features from data, without using language specific resources. So our hybrid neural network model incorporates both bidirectional LSTM and CNN to capture sequential and structural information from text. Our main insight from is that, the sentence level event extraction is not effective in case of news documents. When we take a news article into consideration, we have noticed that the detected events may have their arguments distributed along the length of the document. Sometimes event arguments and corresponding event trigger may appear in two different sentences. And it is evident that the event span(portion of the document where event related information exist) may be a single sentence or multiple sentence or even a whole document. So, identifying events on sentence level will narrow down the event span which leads to a substantial loss in acquired event related information. Eventually the sentence level event extraction system is inadequate to get the document level view of the events. To cope up with this problem, detection and aggregation of coreferring event mentions could be an useful remedy. As future work we would like to explore more sophisticated models for Event extraction and try to incorporate background knowledge and other language resources to improve the system performance.

References

1. Nguyen, Thien Huu, Kyunghyun Cho, and Ralph Grishman. "Joint event extraction via recurrent neural networks." Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016.

2. Chen, Yubo, et al. "Event extraction via dynamic multi-pooling convolutional neural networks." Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Vol. 1. 2015
3. Li, Qi, Heng Ji, and Liang Huang. "Joint event extraction via structured prediction with global features." Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2013.
4. Hong, Yu, et al. "Using cross-entity inference to improve event extraction." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.
5. Nguyen, Thien Huu, and Ralph Grishman. "Event detection and domain adaptation with convolutional neural networks." Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). Vol. 2. 2015.
6. Liu, Shulin, et al. "Exploiting argument information to improve event detection via supervised attention mechanisms." Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2017.
7. Feng, Xiaocheng, Bing Qin, and Ting Liu. "A language-independent neural network for event detection." Science China Information Sciences 61.9 (2018): 092106.
8. Peng, Haoruo, Yangqiu Song, and Dan Roth. "Event detection and co-reference with minimal supervision." Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016.
9. Spiliopoulou, Evangelia, Eduard Hovy, and Teruko Mitamura. "Event detection using frame-semantic parser." Proceedings of the Events and Stories in the News Workshop. 2017.
10. Imran, Muhammad, et al. "Practical extraction of disaster-relevant information from social media." Proceedings of the 22nd International Conference on World Wide Web. ACM, 2013.
11. Imran, Muhammad, et al. "AIDR: Artificial intelligence for disaster response." Proceedings of the 23rd International Conference on World Wide Web. ACM, 2014.
12. Imran, Muhammad, et al. "Extracting information nuggets from disaster-related messages in social media." Isram. 2013.
13. Chowdhury, Soudip Roy, et al. "Tweet4act: Using incident-specific profiles for classifying crisis-related messages." ISCRAM. 2013.
14. Chambers, Nathanael, and Dan Jurafsky. "Unsupervised learning of narrative event chains." Proceedings of ACL-08: HLT (2008): 789-797.
15. Chambers, Nathanael, and Daniel Jurafsky. "A Database of Narrative Schemas." LREC. 2010.
16. Pichotta, Karl, and Raymond J. Mooney. "Learning Statistical Scripts with LSTM Recurrent Neural Networks." AAAI. 2016.
17. Exner, Peter, and Pierre Nugues. "Using semantic role labeling to extract events from Wikipedia." Proceedings of the workshop on detection, representation, and exploitation of events in the semantic web (DeRiVE 2011). Workshop in conjunction with the 10th international semantic web conference. 2011.
18. Kuila, Alapan, and Sudeshna Sarkar. "An Event Extraction System via Neural Networks." FIRE (Working Notes). 2017