Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 2241–2250
Marseille, 11–16 May 2020

A Platform for Event Extraction in Hindi

Sovan Kumar Sahoo, Saumajit Saha, Asif Ekbal, Pushpak Bhattacharyya
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India
{sovan.pcs17, saumajit.mtmc17, asif, pb}@iitp.ac.in

Abstract

Event Extraction is an important task in the widespread field of Natural Language Processing (NLP). Though this task is adequately addressed in English with sufficient resources, we are unaware of any benchmark setup in Indian languages. Hindi is one of the most widely spoken languages in the world. In this paper, we present an Event Extraction framework for Hindi language by creating an annotated resource for benchmarking, and then developing deep learning based models to set as the baselines. We crawl more than seventeen hundred disaster related Hindi news articles from the various news sources. We also develop deep learning based models for Event Trigger Detection and Classification, Argument Detection and Classification and Event-Argument Linking.

Keywords: Event Extraction, Event Trigger, Arguments, Event-Argument Linking

1. Introduction

With the massive growth of electronics technologies and a plethora of digital platforms, an enormous amount of digital content is uploaded every day on the internet. Propelled by this technological advancement, the scope of journalism has also expanded. Nowadays all the leading news agencies have an online version of their newspapers which allow them to publish regular updates on any current event almost instantly. However, extracting relevant information manually about any event from this large amount of data is a near-impossible task. Event extraction, which is an essential task in Information Extraction (IE) deals with this challenge by developing tools and techniques to mine the required information from such unstructured data. Event is something that happens in a specific place at a time or during a particular interval of time. Arguments are the mentions which are involved in any particular event like location, time, participants involved, and so on. Event trigger and argument trigger refer to any word or phrase which describe an event and argument, respectively.

Event trigger detection, event trigger classification, argument trigger detection, argument type classification and event-argument linking are the crucial sub-tasks of a typical event extraction system. In event trigger detection task, word or group of words or phrase which indicates an occurrence of an event are detected, whereas in event trigger classification, the task is to classify the detected event triggers into predefined event types. Similarly, in argument trigger detection, the word or group of words or phrase which indicate an argument of an event are identified, and the motive of argument type classification is to classify each argument trigger into predefined argument roles.

The research community has attempted these problems with utmost attention. But trigger detection and classification are not sufficient for any event extraction system because there may be multiple event triggers of various types in a particular document. Therefore the linking of a particular argument with its corresponding event is very much important. However, compared to the other sub-tasks, event-argument linking has not been addressed much in the literature. In this paper, we address all the above tasks, including event-argument linking to present a robust benchmark setup for event extraction in Hindi. Most of the prior works in event extraction focus on English as there is sufficient resources in this language; however, adequate resources are still missing in low-resource languages like Hindi, as in our case. We attempt to bridge this gap so that the resource created by us can be used as a starting point for further research towards this direction. We crawl disaster-related news data from various web sites. The primary motivation behind choosing "Disaster" as our focus domain is its practical usage in situations of crisis.

2. Related Studies

Being one of the essential tasks in Natural Language Processing (NLP), Event Extraction has already been studied extensively by various researchers across the globe for a long time. However, most of the studies emphasize on sub-tasks like detection and classification of both event and argument triggers. Both feature-based, as well as neural network-based approaches, were tried and tested by the research community. Some of the feature-based approaches have decomposed the entire event extraction task into several sub-tasks and solved them separately (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011). But independent learning of several sub-task leads to error propagation. Some researchers (Riedel and McCallum, 2011a; Riedel and McCallum, 2011b; Li et al., 2013; Venugopal et al., 2014) propose joint event extraction algorithms to deal with this error propagation. Both of the above methods need feature engineering and utilization of the existing NLP tool-kits and resources for doing the same.

In contrast, a neural network can learn those features automatically. Due to this reason, neural based approach gained huge popularity in event extraction task like all the other field of NLP. It includes using a Convolutional Neural Network (CNN) for automatic feature extraction (Nguyen and Grishman, 2015a; Chen et al., 2015; Nguyen and Grishman, 2016). Some authors use Recurrent Neural Network (RNN) (Ghaeini et al., 2016) and the combination of CNN and Bi-LSTM (Feng et al., 2018b) for Event detection. Like feature-based methods in previous cases, neural-based methods also suffer from error propagation if addressed separately. So (Nguyen et al., 2016; Yang and Mitchell, 2016; Liu et al., 2018b) introduced joint models for the Event Extraction task. To boost the performance further Chen et al. (2017) uses *FreeBase* and Liu et al. (2016) uses *FrameNet* to obtain more available data. Liu et al. (2017) proposes to use the annotated argument information explicitly for this task. In recent years (Liu et al., 2018a) make use of a cross language attention model for event detection. Orr et al. (2018) also uses attention mechanism to combine both temporal structure along with syntactic information. (Sha et al., 2018; Nguyen and Grishman, 2018) proposed to use dependency relationships to perform event detection.

Event extraction task has also been assessed in dedicated track in the Text Analysis Conference (TAC). Some of the existing works in Event extraction in disaster domain are reported in (Tanev et al., 2008; Yun, 2011; Klein et al., 2013; Dittrich and Lucas, 2014; Nugent et al., 2017; Burel et al., 2017). However, all the above are mostly in English language. Any significant attempt to build event extraction systems in Indian languages have started in recent times (SharmilaDevi et al., 2017; Sristy et al., 2017; Kuila and Sarkar, 2017; Singh et al., 2017).

As we have mentioned earlier that event-argument linking is less studied in the literature, this task can be related to relation extraction where the relations between a pair of entities are extracted in contrast to our case where relation between event trigger and argument trigger is extracted. Deep Neural Network is being used for relation extraction in recent time.

Zeng et al. (2014) proposed to use CNN in relation extraction for the first time where CNN was used for lexical and sentence level feature extraction. In this paper the authors proposed a novel Position Embedding (PE) feature which was very helpful to achieve high accuracy in classification. PE is also used in (Nguyen and Grishman, 2015b; Santos et al., 2015). Santos et al. (2015) proposed a CNN based relation classifier that performs classification using a Ranking CNN (CR-CNN). Their proposed pairwise ranking loss function makes it easy to reduce the impact of artificial classes. Nguyen and Grishman (2015b) used multiple window sized filters in their CNN architecture compared to single window sized filters as in (Zeng et al., 2014), to capture wider ranges of n-grams. Moreover, their proposed method avoids usage of external resources. Xu et al. (2015a) propose to extract features from the Shortest Dependency Path (SDP) using CNN to avoid irrelevant words. A multi-level attention CNN is proposed by Wang et al. (2016) to detect more subtle cues for relation classification in heterogeneous context. Their proposed method automatically learns which parts are relevant for a given classication. Thus, their proposed method gives best results without any external help. Though CNN is a good feature extractor, it fails to extract syntax as well as hierarchical information of sentence. Based on this observation, Li et al. (2017) introduced hierarchical layers and dependency embedding to CNN architecture. Recently Ren et al. (2018) proposed a CNN based method with Adversarial training (Goodfellow et al., 2014) to enhance the robustness of the classifier. CNN also fails to capture long distance relationship.

Zhang and Wang (2015) proposed a Recurrent Neural Network (RNN) to capture long distance relationships. A SDP based Long Short Term Memory network (SDP-LSTM) is proposed in Xu et al. (2015b) to pick useful information along SDP for different information channel. Zhang et al. (2018a) used the attention layer to capture word-level context information and tensor layer to capture complex connection between the two entities, respectively, on the top of Bi-LSTM. Faruqui and Kumar (2015) proposed a pipe-lined model to develop relation extraction system for any source language. Lin et al. (2017) proposed a multi-lingual attention-based neural relation extraction (MNRE) model. This model employs mono-lingual attention to select the informative sentences within each language. To take the advantages of pattern consistency and complementarity among languages, the proposed model employs cross-lingual attention. Miwa and Bansal (2016) presented a novel end-to-end neural model to extract entities and relations between them. Their proposed model allows joint modeling of both entities and relations using both bidirectional sequential and bidirectional tree-structured LSTM-RNNs. Some researchers used both RNN and CNN to capture local features as well as long term relationships (Cai et al., 2016; Zhang et al., 2018b). Apart from CNN and RNN, some other approaches are also reported in the literature. He et al. (2018) proposed a syntax-aware entity embeddings based on tree-GRU for neural relation classification. Reinforcement learning is used by Feng et al. (2018a) for relation classification from noisy data. Inspired by Generative Adversarial Networks (GANs), Zeng et al. (2018a) proposed a GAN-based method for distant supervised relation extraction. Reinforcement learning has also been explored in Zeng et al. (2018b) to learn sentence relation extractor with the distant supervised dataset. Relation extraction using deep learning techniques in cross-lingual setup has also been explored by the research community. An adversarial multi-lingual neural relation extraction model is proposed by Wang et al. (2018). Recently Subburathinam et al. (2019) investigated a cross-lingual structure transfer method for event and relation extraction using Graph Convolutional Network (GCN). Li et al. (2019) demonstrates a multilingual knowledge extraction system which can perform event extraction, relation extraction, entity discovery, entity linking and coreference. Joint models for event and relation extraction are also proposed by some of the researchers like (Wadden et al., 2019; Han et al., 2019).

3. Motivation and Contribution

From the above discussion, it is clear that event extraction as a field is essential and popular among the research community, but this has been mostly carried out for the resource-rich languages like English. However, there has been a very little works in the low-resource scenario, especially for the Indic languages. Moreover, event extraction in the disaster domain can be very much helpful as we can mine the vast amount of online news and extract crucial information from these, and inform to the various stakeholders ranging from the government agencies to non-government organizations who play a significant role in disaster situations and post-disaster management. Motivated by both resource scarcity and social opportunity mentioned above, we present a benchmark framework for the following sub-tasks in event extraction. This can be both helpful to the society and act as a starting point in this direction in Hindi. We describe the sub-tasks with the help of the following examples.

Example 1:

- **Input Sentence :** भारत में तूफान के खबरों के बीच पाकिस्तान और अफगानिस्तान में भूकंप की खबर है।
- **Transliteration :** bhaarat me toophaan ke khabaro ke beech paakistaan aur aphagaanistaan mein bhookamp kee khabar hai.
- **Translation :** There are reports of earthquakes in Pakistan and Afghanistan amidst news of the storm in India.

Example 2:

- **Input Sentence :** हिमाचल प्रदेश के शिमला में ओलावृष्टि के साथ तेज आंधी आई तथा राज्य के ऊपरी इलाकों में हिम तूफान हुआ।
- **Transliteration :** himaachal pradesh ke shimala mein olaavrshti ke saath tej aandhee aaee tatha raajy ke ooparee ilaakon mein him toophaan hua.
- **Translation :** Heavy thunderstorm accompanied with hailstorm hits Shimla in Himachal Pradesh and snowstorm hit the upper reaches of the state.

Task Description :

- **Event Trigger Detection** : The word or phrase which clearly expresses the occurrence of an event is called an event trigger. Event trigger detection refers to detecting those words or phrases. This task is a sequence labeling task. In the first example, 'तूफान' (toophaan/storm) and 'भूकंप'(bhookamp/earthquake) are the event triggers, whereas in the second example, 'ओलावृष्टि' (olaavrshti /hailstorm), 'आंधी' (aandhee/thunderstorm) and "हिम तूफान" (himtoophaan/snowstorm) are the event triggers.
- **Event Trigger Classification** : It refers to the classification of each event trigger into one of the predefined types. In our current first example, event triggers 'तूफान'(toophaan/storm) and 'भूकंप' (bhookamp/earthquake) belong to *Storm* and *Earthquake* event types, respectively. In our second example, event triggers 'ओलावृष्टि' (olaavrshti /hailstorm), 'आंधी' (aandhee/thunderstorm) and "हिम तूफान" (himtoophaan/snowstorm) belong to *Hail_Storm*, *Storm* and *Blizzard* event types, respectively.
- **Argument Detection** : Arguments are entities, temporal expressions, or values that act as attributes or participants with some predefined roles of an event. Here the task is to detect such trigger words. For example, in our first case, 'भारत' (bhaarat/India) and "पाकिस्तान और अफगानिस्तान" (paakistaan aur aphagaanistaan/Pakistan and Afghanistan) are the arguments. In our second example, "हिमाचल प्रदेश के शिमला" (himaachal pradesh ke shimala/Shimla in Himachal Pradesh) and "राज्य के ऊपरी इलाकों" (raajyke ooparee ilaakon/upper reaches of the state) are the arguments.
- **Argument Classification** : This task refers to the classification of each argument into the predefined argument roles. In our first example, 'भारत'(bhaarat/India) and "पाकिस्तान और अफगानिस्तान" (paakistaan aur aphagaanistaan/Pakistan and Afghanistan) are to be classified as *Place* arguments. In our second example also, "हिमाचल प्रदेश के शिमला" (himaachal pradesh ke shimala/Shimla in Himachal Pradesh) and "राज्य के ऊपरी इलाकों" (raajyke ooparee ilaakon/upper reaches of the state) are also to be classified as *Place* arguments.
- **Event-argument Linking** : This corresponds to linking argument triggers with its corresponding event triggers. In our current work, we formulate it as a classification problem, which means given a pair of event and argument trigger, the task is to find whether there is any link between them or not. In our first example, 'Storm' (तूफान) happened in 'India' (भारत), whereas 'Earthquake' (भूकंप) happened in "Pakistan and Afghanistan" (पाकिस्तान और अफगानिस्तान), so we need to link 'Storm' with 'India' and 'Earthquake' with "Pakistan and Afghanistan". Similarly in our second example, we need to link 'Thunderstorm' (आंधी) and 'Hailstorm' (ओलावृष्टि) with "Shimla in Himachal Pradesh" (हिमाचल प्रदेश के शिमला) and 'Snowstorm' (हिम तूफान) with "upper reaches of the state" (राज्य के ऊपरी इलाकों).

4. Benchmark Setup

Creating a benchmark setup[1] is a very challenging task. The whole process is divided into multiple sub-processes like creating a web crawler, crawling raw data, cleaning and pre-processing, and annotation. In subsequent subsections, we discuss all the sub-processes in details.

4.1. Data Crawling

We design a web crawler which crawls news data from multiple online news portals[2]. We crawled more than 1700 news articles from the Internet. We have 28 event types which belong to one of the two main categories, namely Natural disaster and Man-made disaster. Each of the crawled articles reports about one of the 28 types of events.

4.2. Pre-processing

All the downloaded raw HTML files are cleaned and raw Hindi texts are extracted form those articles. After that each

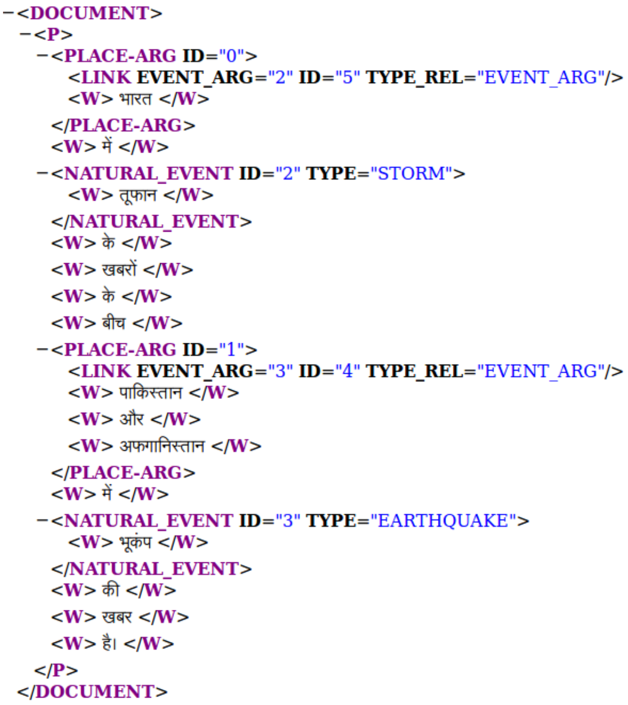
[1] The resource is available at `https://www.iitp.ac.in/~ai-nlp-ml/resources.html#EventExtraction`

[2] List of few news portals : www.bhaskarhindi.com, www.amarujala.com, www.jansatta.com, www.inextlive.com, www.bhaskar.com, www.patrika.com, www.naidunia.com, www.jagran.com

of the articles are converted into XML formats. This XML files are used for annotation.

4.3. Data Annotation

After pre-processing, we annotated all the documents following TAC KBP[3] annotation guidelines for our annotation task. Three annotators, with expertise in linguistics, were employed for the annotation task. All the annotators have post-graduate level knowledge and possess good proficiency in Hindi. The annotation of each document includes marking the event triggers along with their corresponding types, the argument triggers along with their corresponding roles, and linking information where all the argument triggers related to each event triggers are linked with it. All the marked triggers and links have unique ID for a particular document. We use PALinkA (Orăsan, 2003) annotation tool for this task. Figure 1 shows the screenshot of annotated XML file. The tag set is organized into an ontology which includes two types of events - *Natural* and *Man-made*, and eleven types of arguments - *Place*, *Time*, *Casualty*, *Reason*, *After_Effect*, *Participant*, *Intensity*, *Magnitude*, *Epicentre*, *Name* and *Speed*. Some of the arguments are common to all events like *Place*, *Time*, *Reason*, and *After_Effect*. Loss of life due to any disaster belongs to *Casualty* argument type, whereas anything that happens after any event, belongs to *After_Effect* argument type. Some of the argument types are specific to any particular event. For example, *Magnitude* and *Epicentre* are related to earthquakes whereas *Intensity* and *Speed* are related to cyclones, storms, tornado, hailstorm etc. *Name* argument is used to name any volcano or cyclone. The ontology consists of three levels, where both *Natural* and *Man-made* disaster types are further divided into different sub-types. We have a total of 28 sub-types of disasters in our Hindi corpus. Sub-types of natural disaster types comprise of *Forest_Fire*, *Blizzard*, *Cold_Wave*, *Heat_Wave*, *Earthquake*, *Cyclone*, *Storm*, *Hail_Storms*, *Hurricane*, *Tornado*, *Avalanches*, *Land_Slide*, *Rock_Fall*, *Floods*, *Tsunami* and *Volcano*. Man-made disaster includes event types like *Armed_Conflicts*, *Riots*, *Shoot_Out*, *Terrorist_Attack*, *Normal_Bombing*, *Surgical_Strikes*, *Transport_Hazards*, *Fire*, *Vehicular_Collision*, *Train_Collision*, *Aviation_Hazard*, *Industrial_Accident*.
We measure the inter-annotator agreement ratio by asking all the three annotators to annotate 5% of total documents. We observe the multi-rater Kappa agreement (Fleiss, 1971) ratio of 0.82, 0.79, 0.69 and 0.67 for event trigger detection, event trigger classification, argument trigger detection and argument trigger classification, respectively. As most of the trigger words are multi-word expression, we calculate Kappa score for partial match. For partial match, we consider most important words only. For example, for an event trigger, "भूकंप के झटके" (Earthquake tremors), if one annotator annotates *B_Event*, *I_Event* and *I_Event* and the other annotator annotates *B_Event*, *O* and *O*, respectively, we consider both as correct agreement because for both the cases, the most important word 'भूकंप' (Earthquake) is annotated as part of event trigger. Multi-rater Kappa agreement ratio for event-argument linking is 0.74.



```
−<DOCUMENT>
  −<P>
    −<PLACE-ARG ID="0">
        <LINK EVENT_ARG="2" ID="5" TYPE_REL="EVENT_ARG"/>
        <W> भारत </W>
      </PLACE-ARG>
      <W> में </W>
    −<NATURAL_EVENT ID="2" TYPE="STORM">
        <W> तूफान </W>
      </NATURAL_EVENT>
      <W> के </W>
      <W> खबरों </W>
      <W> के </W>
      <W> बीच </W>
    −<PLACE-ARG ID="1">
        <LINK EVENT_ARG="3" ID="4" TYPE_REL="EVENT_ARG"/>
        <W> पाकिस्तान </W>
        <W> और </W>
        <W> अफगानिस्तान </W>
      </PLACE-ARG>
      <W> में </W>
    −<NATURAL_EVENT ID="3" TYPE="EARTHQUAKE">
        <W> भूकंप </W>
      </NATURAL_EVENT>
      <W> की </W>
      <W> खबर </W>
      <W> है। </W>
    </P>
  </DOCUMENT>
```

Figure 1: Screenshot of annotated XML file

4.4. Important Aspects of Data Annotation

At the time of annotating the data, we face two important aspects which should be taken care of during system building. We describe those aspects through the following example.

Example 3

- **Input Sentence :** दक्षिण कश्मीर के कुलगाम में आतंकी हमला हुआ है। पांच बाहरी मजदूरों के मारे जाने की खबर है।
- **Transliteration :** dakshin kashmeer ke kulagaam mein aatankee hamala hua hai. paanch baaharee majadooron ke maare jaane kee khabar hai.
- **Translation :** There has been a terrorist attack in Kulgam in South Kashmir. There are reports of five outsider workers being killed.

The important aspects, along with the corresponding challenges, are described below.

- **Multi-word triggers :** Multi-word triggers are frequently observed. In example 3, we observe that "दक्षिण कश्मीर के कुलगाम" (Kulgam in South Kashmir) and "पांच बाहरी मजदूरों" (five outsider workers) are multi-word argument triggers, whereas "आतंकी हमला" (terrorist attack) is a multi-word event trigger. Multi-word trigger detection is a very crucial task and needs special attention to get solved (Ghaeini et al., 2016). In the above example, "आतंकी हमला" (terrorist attack) is a multi-word event trigger with annotation labels *B_Event* and *I_Event* respectively. On the other hand 'हमला' (attack) itself is an event trigger with annotation label *B_Event*. So, for the first case, the correct detection label for 'हमला' (attack) should be *I_Event*, and *B_Event* for the later case.

[3] https://www.nist.gov/tac/

- **Inter-sentence Event-argument link :** It is often observed in Hindi news items that journalists write smaller sentences while reporting any events. For this reason, events and their corresponding arguments span across multiple sentences. In our following example, we can see that the event trigger "आतंकी हमला" (terrorist attack) is in first sentence and one of its arguments (*Casualty*) "पांच बाहरी मजदूरों" (five outsider workers) is in the next sentence. The difficulty level of linking event trigger with its corresponding argument trigger increases with the increasing distance between them.

4.5. Dataset Statistics

Table 1 shows the dataset statistics of our resources. We have a total of 1709 number of annotated documents. Average number of sentences per document is 18.5. Average length of sentences *i.e.* average number of words per sentence is 15.5. Average number of words per event trigger and argument trigger are 1.67 and 4.7, respectively. We randomly split our whole dataset file-wise into Train, Dev, and Test sets in 13:3:4 ratio. We split our dataset file-wise rather than sentence-wise so that Train, Dev, and Test data for all the sub-tasks can be used from the same corresponding distribution, respectively.

Features	**Statistics**
Total number of document annotated	1,709
Total Number of sentences	31,650
Average sentences per document	18.5
Average number of words per sentence	15.5
Total number of Event Triggers	10,383
Average length of Event Triggers	1.67
Total number of Argument Triggers	25,189
Average length of Argument Triggers	4.7

Table 1: Dataset statistics

5. Evaluation

Event and Argument Trigger Classification : We formulate both event trigger detection and argument detection as sequence labelling problems. We also argue that they can be solved jointly as both of these tasks are co-related. Thus, we can define the task as : Given a sentence of the form w_1,w_2,w_3, ... , w_n, the task is to predict the event and argument labels (l_i) for each word (w_i), where l_i∈ {I,O,B}[4]. We develop two baseline models based on Bi-LSTM+CRF (Huang et al., 2015; Lample et al., 2016) and Bi-LSTM+Softmax. Bi-LSTM (Schuster and Paliwal, 1997) is a very good sentence encoder and CRF (Conditional Random Field) (Lafferty et al., 2001) is very efficient in sequence labeling as it uses state transition matrix to take care of the past and future tags to predict the current tag (Sutton et al., 2012). Sequence labeling problem can also be formulated as multi-class classification problem as in (Chen et al., 2015; Ghaeini et al., 2016). To represent each word as vector, we use *fastText* word embeddings of dimension 300. The pre-trained monolingual word vectors are downloaded from *fastText*[5]. Before training the baseline model, we perform the pre-processing operation on the annotated XML files to generate an input-output sequence for training. Table 2 depicts a sample pre-processing output of Example 1, shown in Section 3.

Words	**Event & Argument Trigger Detection**	**Event & Argument Classification**
भारत	B_Arg	Place
में	O	None
तूफान	B_Event	Storm
के	O	None
खबरो	O	None
के	O	None
बीच	O	None
पाकिस्तान	B_Arg	Place
और	I_Arg	Place
अफगानिस्तान	I_Arg	Place
में	O	None
भूकंप	B_Event	Earthquake
की	0	None
खबर	0	None
है	0	None

Table 2: Sample annotation for the sentence given in Example-1 in Section 3

Feature	**Train**	**Dev**	**Test**
# documents	1,127	259	323
# sentences	20,735	4,682	6,233
# Event Trigger words	11,340	2,649	3,284
# Argument Trigger words	77,069	18,269	22,719

Table 3: Dataset Statistics for event and argument detection

Features	**Train**	**Dev**	**Test**
# of Event Triggers	6,815	1,570	1,981
# of Arguments Triggers	16,314	3,930	4,945

Table 4: Dataset Statistics for event and argument classification

Features	**Train**	**Dev**	**Test**
# of files	1,127	259	323
# of instances	23,226	5,591	7,127
# of YES cases	12,480	2,881	3,773
# of No cases	10,746	2,710	3,354

Table 5: Dataset Statistics for event-argument linking

[4] The encoding scheme is according to IOB2, where I indicates the word tokens that appear inside the trigger, B denotes the beginning of a trigger and O denotes the outside of an event trigger or argument trigger. The B is used only when two event or argument triggers of the same type appear in consecutive sequence (Ramshaw and Marcus, 1999)

[5] `https://fasttext.cc`

Event-Argument pair	Class labels
भारत (India), तूफान (storm)	1
भारत (India), भूकंप (earthquakes)	0
पाकिस्तान और अफगानिस्तान (Pakistan and Afghanistan), तूफान (storm)	0
पाकिस्तान और अफगानिस्तान (Pakistan and Afghanistan),भूकंप (earthquakes)	1

Table 6: Training instances generated from Example 1

Event and Argument Trigger Classification : We develop baseline models for event trigger as well as argument trigger classification. Given an event or argument trigger and its' both side context, the task is to classify the event trigger or argument trigger into the predefined event types or argument types, respectively. In our previous example 3, "दक्षिण कश्मीर के कुलगाम " (Kulgam in South Kashmir) and "पांच बाहरी मजदूरों" (five outsider workers) to be classified as *Place* argument and *Casualty* argument, respectively. "आतंकी हमला" (terrorist attack) is an event trigger which should be classified as *Terrorist_Attack* event type. We observe that, contextual information helps to improve the classification performance. In example 3, "पांच बाहरी मजदूरों" (five outsider workers) could be miss-classified as *Participant* but the context "मारे जाने की खबर है" (being killed) helps to classify it as *Casualty*. For classification tasks (event and argument), we use Bi-LSTM for encoding both triggers (event or argument) and contexts (left and right) of the respective triggers. The output of final hidden layers (event/argument and context of both sides) are concatenated and passed through a dense layer for classification. We use *Softmax* function for classification.

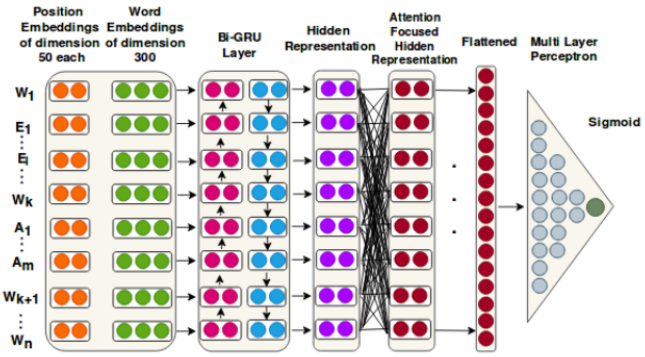


Figure 2: Baseline 3 for Event-Argument Linking : Bi-GRU + Self-attention

Event-Argument Linking : We frame Event-Argument linking as a binary classification problem. Given a sentence, comprising a sequence of words, w_1, w_2, e_1, e_2, e_i, w_3, ... , w_k, a_1, a_2, a_j, w_{k+1}, ... , w_n, where e_i and a_j denote an event trigger and an argument trigger, respectively. The task is to predict whether there exists any link between the event trigger e_i and the argument trigger a_j.

The first example of Section 3 has two events triggers, namely 'Storm' (तूफान) and 'Earthquake' (भूकंप) and two *Place* arguments, namely 'India' (भारत) and "Pakistan and Afghanistan" (पाकिस्तान और अफगानिस्तान). It results in a total of four event-argument pairs. Thus, we create four training instances, one for each event-argument pair. As the 'Storm' (तूफान) happened in 'India' (भारत), so we assign the classification label as 1 (*Yes*), whereas the 'Storm' (तूफान) did not happen in "Pakistan and Afghanistan" (पाकिस्तान और अफगानिस्तान), so we assign the classification label as 0 (*No*).

We also perform event-argument linking across multiple sentences. As mentioned earlier, there is a considerable number of examples where an event trigger is in one sentence, and the arguments are in next or next to the next sentence. For our current experiments, we consider maximum two consecutive sentences, across which event and its corresponding arguments are spanned. Table 6 depicts the possible event-argument pair and their corresponding classification labels.

We develop three baseline models. The first model is a CNN based model, where output of CNN layer is passed through a Multi-Layer Perceptron (MLP) for classification. The second model is a Bi-GRU+CNN based model, which takes care of the bidirectional relationship. The vector representation of input passes through a Bi-GRU layer, which captures the two-way relationship between the event-argument. The output of the Bi-GRU layer passes through a CNN layer. CNN tries to extract convoluted local features. The output of CNN is fed into a MLP layer followed by a *Sigmoid* activation function for binary classification. Our third baseline model also uses Bi-GRU layer for the same reason as our previous model, however in this case a self-attention layer is added on top of the Bi-GRU layer. The resulting output of the self-attention layer is then fed to a MLP layer for the final output.

Figure 2 shows the architecture diagram of our third baseline (Bi-GRU+Self-attention) model. Each word w_i of the input sentence S_i = (w_1, w_2, . . . , w_n) is represented by the concatenation of two types of embeddings: (i). word embeddings (WE) which capture the syntax and semantics of the word. For word embedding, we use *fastText* word embeddings. (ii). a position embedding (PE) which identifies both the target event and argument trigger of our interest. The PE also identifies the nearness of each word with respect to the target event and argument word or phrases. The input sentence S_i is the sequence of vectors S_i = (w_1, w_2, . . . , w_n), where $w_i \in R^d$ and d = d_w + 2×d_p. d_w and d_p are the dimensions of word embedding and position embedding, respectively. We set the maximum length of each input sentence to be 100. Sentences having shorter lengths are padded up using a fixed random vector of the same dimension as d_e. Longer sentences are truncated till the maximum length.

5.1. Results and Discussions

The experimental results for event and argument detection of this task are shown in Table 8. We observe that Bi-LSTM+Softmax performs better than Bi-LSTM+CRF model specifically in detecting the beginning of event trigger (*B_Event*). The reason behind the lower precision of the model is due to its incorrect detection of many words as *B_Event*, which mostly belong to *B_Arg* and *I_Arg*. Table 3 shows the dataset statistics for event and argument detection. For trigger detection task, we report *precision*

Event Types	F1-Score with context	F1-Score without context	Support Count
Storm	**0.71**	0.69	118
Tornado	**0.67**	0.61	35
Cyclone	**0.36**	0.34	39
Hurricane	**0.59**	0.10	35
Hail_Storms	**0.96**	**0.96**	40
Blizzard	0.88	**0.92**	21
Cold_Wave	0.97	**0.98**	50
Heat_Wave	0.89	**0.90**	29
Avalanches	0.96	**1**	37
Land_Slide	**0.88**	0.85	73
Earthquake	**0.99**	**0.99**	236
Tsunami	0.89	**0.92**	7
Floods	0.96	**0.97**	58
Forest_Fire	**0.82**	0.33	48
Rock_Fall	**0.44**	0.39	6
Volcano	**0.96**	0.82	49
Terrorist_Attack	**0.69**	0.67	118
Suicide_Attack	**0.76**	0.69	121
Normal_Bombing	0.10	**0.32**	7
Armed_Conflicts	**0.62**	0	8
Shoot_Out	**0.82**	0.77	122
Riots	**0.83**	0.8	97
Fire	**0.85**	0.78	206
Industrial_Accident	**0.49**	0.2	65
Aviation_Hazard	**0.74**	0.59	76
Train_Collision	**0.38**	0.32	82
Transport_Hazard	**0.38**	0.28	66
Vehicular_Collision	**0.75**	0.55	130

Table 7: Experimental Results for event trigger classification

	Bi-LSTM+CRF			Bi-LSTM+Softmax		
	P	**R**	**F1**	**P**	**R**	**F1**
B-Event	0.23	0.73	0.35	0.63	0.73	**0.68**
I-Event	0.62	0.45	**0.52**	0.65	0.43	0.51
B-Arg	0.62	0.49	**0.55**	0.56	0.53	0.54
I-Arg	0.65	0.43	0.52	0.59	0.54	**0.57**
O	0.98	0.99	0.99	0.99	0.99	0.99

Table 8: Experimental results for event and argument detection. Here, P stands for precision, R stands for recall and F1 stands for F1-score

(P), *recall (R)* and *F1-Score(F1)*. Table 7 and Table 9 show the results of event and argument trigger classification results, respectively. Here, support count refers to the number of event and argument triggers for each type of event and argument, respectively. The results depict that contextual information helps in most of the cases for event and argument classification, respectively in terms of *F1-Score*.

For event-argument linking also, we split our entire data into train, test and validation set for our experiments. We also perform five-fold cross validation and report results of each fold. We observe that our second baseline model (BI-GRU+CNN) performs slightly better than the third baseline model (Bi-GRU+Self-attention) and outperforms the first model by a significant margin. The dataset statistics for these experiments are shown in Table 5. Table 10 depicts the results of our experiments in terms of *F1-Score*.

Argument Types	F1-Score with context	F1-Score without context	Support Count
Time	**0.98**	**0.98**	666
Place	**0.96**	0.95	1577
Casualties	**0.90**	0.89	79
After_Effect	**0.90**	0.87	969
Reason	**0.71**	0.59	160
Participant	**0.89**	0.88	494
Intensity	**0.79**	0.62	99
Epicentre	**0.78**	0.38	31
Magnitude	**0.95**	0.89	56
Name	**0.90**	0.75	85
Speed	0.70	**0.74**	10

Table 9: Experimental results for argument classification

Tests	Bi-GRU + Self-attention		CNN		Bi-GRU + CNN	
	YES	**NO**	**YES**	**NO**	**YES**	**NO**
Train -Test	**0.87**	**0.85**	0.82	0.79	0.86	0.84
Fold 1	0.88	**0.87**	0.84	0.82	**0.89**	**0.87**
Fold 2	**0.88**	0.86	0.84	0.82	**0.88**	**0.87**
Fold 3	**0.88**	0.86	0.85	0.82	**0.88**	**0.87**
Fold 4	**0.89**	0.86	0.84	0.82	**0.89**	**0.88**
Fold 5	0.88	**0.87**	0.84	0.82	**0.89**	0.86

Table 10: Experimental Results for event-argument linking

6. Conclusion

In this paper, we have proposed a benchmark setup of event extraction in Hindi. We have crawled news articles from various Hindi news portals. We have performed cleaning, pre-processing, and converted the cleaned files to the XML files and annotated as per the annotation guidelines. The annotated document consists of annotation for five sub-tasks of Event extraction, namely Event trigger detection, Event Trigger Classification, Argument detection, Argument role classification, and Event-argument linking. We develop deep learning based baseline models for Event and Argument Trigger detection and Event-argument linking. Experimental results show promising results in terms of *F1-Score*. In future, we would like to investigate Event Realis status classification and Event coreference resolution.

7. Acknowledgement

The research reported in this paper is an outcome of the project titled “A Platform for Cross-lingual and Multilingual Event Monitoring in Indian Languages”, supported by IMPRINT-1, MHRD, Govt. of India, and MeiTY, Govt. of India. Sovan Kumar Sahoo gratefully acknowledges

“Visvesvaraya PhD Scheme for Electronics and IT”, under the Ministry of Electronics and Information Technology, Government of India. Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia). We would also like to acknowledge linguists Swati Tiwari (IIT Patna), Monalisa Bhattacherjee (IIT Patna) and Jaya Jha (IIT Bombay) for contributing in annotation of resources.

8. Bibliographical References

Burel, G., Saif, H., Fernandez, M., and Alani, H. (2017). On semantics and deep learning for event detection in crisis situations. *Workshop on Semantic Deep Learning (SemDeep), at ESWC 2017, 29 May 2017, Portoroz, Slovenia*.

Cai, R., Zhang, X., and Wang, H. (2016). Bidirectional Recurrent Convolutional Neural Network for Relation Classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 756–765.

Chen, Y., Xu, L., Liu, K., Zeng, D., and Zhao, J. (2015). Event extraction via dynamic multi-pooling convolutional neural networks. In 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), volume 1, pages 167–176.

Chen, Y., Liu, S., Zhang, X., Liu, K., and Zhao, J. (2017). Automatically labeled data generation for large scale event extraction. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 409–419.

Dittrich, A. and Lucas, C. (2014). Is this twitter event a disaster? *AGILE Digital Editions*.

Faruqui, M. and Kumar, S. (2015). Multilingual open relation extraction using cross-lingual projection. *arXiv preprint arXiv:1503.06450*.

Feng, J., Huang, M., Zhao, L., Yang, Y., and Zhu, X. (2018a). Reinforcement Learning for Relation Classification from Noisy Data. In Proceedings of AAAI, pages 5779–5786.

Feng, X., Qin, B., and Liu, T. (2018b). A language-independent neural network for event detection. *Science China Information Sciences*, 61(9):092106.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Ghaeini, R., Fern, X., Huang, L., and Tadepalli, P. (2016). Event nugget detection with forward-backward recurrent neural networks. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 369–373.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Han, R., Ning, Q., and Peng, N. (2019). Joint event and temporal relation extraction with shared representations and structured prediction. *arXiv preprint arXiv:1909.05360*.

He, Z., Chen, W., Li, Z., Zhang, M., Zhang, W., and Zhang, M. (2018). SEE: Syntax-aware Entity Embedding for Neural Relation Extraction. *arXiv preprint arXiv:1801.03603*.

Hong, Y., Zhang, J., Ma, B., Yao, J., Zhou, G., and Zhu, Q. (2011). Using cross-entity inference to improve event extraction. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pages 1127–1136. Association for Computational Linguistics.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. *Proceedings of ACL-08: HLT*, pages 254–262.

Klein, B., Castanedo, F., Elejalde, I., López-de Ipina, D., and Nespral, A. P. (2013). Emergency event detection in twitter streams based on natural language processing. In International Conference on Ubiquitous Computing and Ambient Intelligence, pages 239–246. Springer.

Kuila, A. and Sarkar, S. (2017). An event extraction system via neural networks. *FIRE (Working Notes)*, pages 136–139.

Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Li, Q., Ji, H., and Huang, L. (2013). Joint event extraction via structured prediction with global features. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 73–82.

Li, B., Zhao, X., Wang, S., Lin, W., and Xiao, W. (2017). Relation Classification using Revised Convolutional Neural Networks. In Systems and Informatics (ICSAI), 2017 4th International Conference on, pages 1438–1443. IEEE.

Li, M., Lin, Y., Hoover, J., Whitehead, S., Voss, C., Dehghani, M., and Ji, H. (2019). Multilingual entity, relation, event and human value extraction. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pages 110–115.

Liao, S. and Grishman, R. (2010). Using document level cross-event inference to improve event extraction. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 789–797. Association for Computational Linguistics.

Lin, Y., Liu, Z., and Sun, M. (2017). Neural relation extraction with multi-lingual attention. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 34–43.

Liu, S., Chen, Y., He, S., Liu, K., and Zhao, J. (2016). Leveraging framenet to improve automatic event detection. In Proceedings of the 54th Annual Meeting of the

Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 2134–2143.

Liu, S., Chen, Y., Liu, K., Zhao, J., et al. (2017). Exploiting argument information to improve event detection via supervised attention mechanisms.

Liu, J., Chen, Y., Liu, K., and Zhao, J. (2018a). Event detection via gated multilingual attention mechanism. In Thirty-Second AAAI Conference on Artificial Intelligence.

Liu, X., Luo, Z., and Huang, H. (2018b). Jointly multiple events extraction via attention-based graph information aggregation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1247–1256.

Miwa, M. and Bansal, M. (2016). End-to-end relation extraction using lstms on sequences and tree structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 1105–1116.

Nguyen, T. H. and Grishman, R. (2015a). Event detection and domain adaptation with convolutional neural networks. In 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), volume 2, pages 365–371.

Nguyen, T. H. and Grishman, R. (2015b). Relation Extraction: Perspective from Convolutional Neural Networks. In Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, pages 39–48.

Nguyen, T. H. and Grishman, R. (2016). Modeling skipgrams for event detection with convolutional neural networks. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 886–891.

Nguyen, T. H. and Grishman, R. (2018). Graph convolutional networks with argument-aware pooling for event detection. In Thirty-Second AAAI Conference on Artificial Intelligence.

Nguyen, T. H., Cho, K., and Grishman, R. (2016). Joint event extraction via recurrent neural networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 300–309.

Nugent, T., Petroni, F., Raman, N., Carstens, L., and Leidner, J. L. (2017). A comparison of classification models for natural disaster and critical event detection from news. In Big Data (Big Data), 2017 IEEE International Conference on, pages 3750–3759. IEEE.

Orăsan, C. (2003). PALinkA: A highly customisable tool for discourse annotation. In Proceedings of the Fourth SIGdial Workshop of Discourse and Dialogue, pages 39–43.

Orr, W., Tadepalli, P., and Fern, X. (2018). Event detection with neural networks: A rigorous empirical evaluation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 999–1004.

Ramshaw, L. A. and Marcus, M. P. (1999). Text chunking using transformation-based learning. In *Natural language processing using very large corpora*. Springer, pp. 157–176.

Ren, F., Zhou, D., Liu, Z., Li, Y., Zhao, R., Liu, Y., and Liang, X. (2018). Neural Relation Classification with Text Descriptions. In Proceedings of the 27th International Conference on Computational Linguistics, pages 1167–1177.

Riedel, S. and McCallum, A. (2011a). Fast and robust joint models for biomedical event extraction. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1–12. Association for Computational Linguistics.

Riedel, S. and McCallum, A. (2011b). Robust biomedical event extraction with dual decomposition and minimal domain adaptation. In Proceedings of the BioNLP Shared Task 2011 Workshop, pages 46–50. Association for Computational Linguistics.

Santos, C. N. d., Xiang, B., and Zhou, B. (2015). Classifying Relations by Ranking with Convolutional Neural Networks. *arXiv preprint arXiv:1504.06580*.

Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Sha, L., Qian, F., Chang, B., and Sui, Z. (2018). Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In Thirty-Second AAAI Conference on Artificial Intelligence.

SharmilaDevi, V., Kannimuthu, S., and Safeeq, G. (2017). Kce_dalab@ eventxtract-il-fire2017: Event extraction using support vector machines. *FIRE (Working Notes)*, pages 144–146.

Singh, J. P., Dwivedi, Y. K., Rana, N. P., Kumar, A., and Kapoor, K. K. (2017). Event classification and location prediction from tweets during disasters. *Annals of Operations Research*, pages 1–21.

Sristy, N. B., Krishna, N. S., and Somayajulu, D. V. (2017). Event extraction from social media text using conditional random fields. In FIRE (Working Notes), pages 140–143.

Subburathinam, A., Lu, D., Ji, H., May, J., Chang, S.-F., Sil, A., and Voss, C. (2019). Cross-lingual structure transfer for relation and event extraction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 313–325.

Sutton, C., McCallum, A., et al. (2012). An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

Tanev, H., Piskorski, J., and Atkinson, M. (2008). Real-time news event extraction for global crisis monitoring. In International Conference on Application of Natural Language to Information Systems, pages 207–218. Springer.

Venugopal, D., Chen, C., Gogate, V., and Ng, V. (2014). Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 831–843.

Wadden, D., Wennberg, U., Luan, Y., and Hajishirzi, H. (2019). Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Wang, L., Cao, Z., De Melo, G., and Liu, Z. (2016). Relation Classification via Multi-level Attention CNNs. In Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers), volume 1, pages 1298–1307.

Wang, X., Han, X., Lin, Y., Liu, Z., and Sun, M. (2018). Adversarial multi-lingual neural relation extraction. In Proceedings of the 27th International Conference on Computational Linguistics, pages 1156–1166.

Xu, K., Feng, Y., Huang, S., and Zhao, D. (2015a). Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling. *arXiv preprint arXiv:1506.07650*.

Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., and Jin, Z. (2015b). Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths. In Proceedings of the 2015 conference on empirical methods in natural language processing, pages 1785–1794.

Yang, B. and Mitchell, T. (2016). Joint extraction of events and entities within a document context. *arXiv preprint arXiv:1609.03632*.

Yun, H.-W. (2011). Disaster events detection using twitter data. *Journal of information and communication convergence engineering*, 9(1):69–73.

Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation Classification via Convolutional Deep Neural Network. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2335–2344.

Zeng, D., Dai, Y., Li, F., Sherratt, R. S., and Wang, J. (2018a). Adversarial learning for distant supervised relation extraction. *Computers, Materials & Continua*, 55(1):121–136.

Zeng, X., He, S., Liu, K., and Zhao, J. (2018b). Large scaled relation extraction with reinforcement learning. In Thirty-Second AAAI Conference on Artificial Intelligence.

Zhang, D. and Wang, D. (2015). Relation Classification via Recurrent Neural Network. *arXiv preprint arXiv:1508.01006*.

Zhang, R., Meng, F., Zhou, Y., and Liu, B. (2018a). Relation Classification via Recurrent Neural Network with Attention and Tensor Layers. *Big Data Mining and Analytics*, 1(3):234–244.

Zhang, X., Chen, F., and Huang, R. (2018b). A Combination of RNN and CNN for Attention-based Relation Classification. *Procedia computer science*, 131:911–917.