A Multi-task Model for Multilingual Trigger Detection and Classification

Sovan Kumar Sahoo[1], Saumajit Saha[2], Asif Ekbal[1], Pushpak Bhattacharyya[1]
[1]Department of Computer Science and Engineering, [2]Department of Mathematics
Indian Institute of Technology Patna
{sovan.pcs17, saumajit.mtmc17, asif, pb}@iitp.ac.in

Abstract

In this paper we present a deep multi-task learning framework for multilingual event and argument trigger detection and classification. In our current work, we identify detection and classification of both event and argument triggers as related tasks and follow a multi-tasking approach to solve them simultaneously in contrast to the previous works where these tasks were solved separately or learning some of the above mentioned tasks jointly. We evaluate the proposed approach with multiple low-resource Indian languages. As there were no datasets available for the Indian languages, we have annotated disaster related news data crawled from the online news portal for different low-resource Indian languages for our experiments. Our empirical evaluation shows that multi-task model performs better than the single task model, and classification helps in trigger detection and vice-versa.

1 Introduction

Event Extraction is an important task in Natural Language Processing (NLP). An event can be an occurrence happening in certain place during a particular interval of time. In text, the word or phrase that describes an event is called event trigger. Argument of an event refers to the attributes such as the location, time of occurrence of the event, participants involved and so on. Therefore event trigger detection, event trigger classification, argument trigger detection and argument trigger classification are the four important sub-tasks of event extraction. In our current paper, we have solved all the four problems using a Multi-task architecture. Multi-task learning (MTL), which essentially means performing more than one *related* task simultaneously, has been proven to be effective for various NLP tasks in recent times (Ruder, 2017). The key idea behind MTL is that the inductive transfer of knowledge, learned for a particular task, can help to improve the performance of another task by means of parameter sharing between tasks. According to Caruana (1997), "MTL improves generalization by leveraging the domain-specific information contained in the training signals of *related* tasks". In our current work, we have identified *detection* and *classification* of both event and arguments as two *related* tasks. As both event and argument trigger detection are sequence labelling problems, we have merged those two sub-tasks into one and used a single loss function. For the same reason, we have merged event and argument trigger classification task into one task and used another loss function. Thus in our proposed architecture, even though we have two main tasks for learning shared representation, we have basically solved four sub-tasks *viz.* event trigger detection, event trigger classification, argument detection and argument classification. Our proposed architecture has two variants which are further discussed later in this paper. As we are working with low-resource languages which have data sparsity issue, we have proposed a *multi-task, multi-lingual* architecture which is trained on both Hindi and Bengali data. Due to unavailability of training data in these two languages, we have annotated disaster related news data crawled from online news portals for our experiments.

2 Related Works

Being a very important problem in NLP, Event Extraction has already been explored by the research community for a long time. Some feature based approaches have decomposed the entire event extraction task into two sub-tasks and solved them separately (Ji and Grishman, 2008; Hong et al., 2011; Liao and Grishman, 2010). But the main problem of this approach is error propagation which is dealt by Riedel and McCallum (2011a), Riedel and McCallum (2011b), Li et al. (2013), Venugopal et al. (2014) using a joint event extraction algorithm. However both of the

D M Sharma, P Bhattacharyya and R Sangal. Proc. of the 16th Intl. Conference on Natural Language Processing, pages 160–169
Hyderabad, India, December 2019.

above approaches have used hand-designed feature. Nguyen and Grishman (2015) propose a Convolutional Neural Network (CNN) for automatic feature extraction. Chen et al. (2015) introduce a dynamic multi-pooling CNN which uses a dynamic multi-pooling layer according to event triggers and arguments in multi-event sentences, to capture more crucial information. In another work, Nguyen and Grishman (2016) propose a skip-gram based CNN model which allows non-consecutive convolution. Ghaeini et al. (2016) propose a forward-backward Recurrent Neural Network (RNN) to detect event triggers which can be in the form of both words or phrases. Feng et al. (2018) propose a language independent neural network which uses both CNN and Bi-LSTM for Event detection. Liu et al. (2016) propose to improve the performance of event detection by using the events automatically detected from FrameNet. Though these neural based systems perform well, they still suffer from error propagation issue. To overcome this issue, Nguyen et al. (2016) propose a joint framework with bidirectional RNN. However Liu et al. (2017) observe that joint model achieves insignificant improvements on event detection task. They analyze the problem of joint models on the task of event detection, and propose to use the annotated argument information explicitly for this task. Yang and Mitchell (2016) also propose a joint model for event and entity extraction but in document level instead of sentence level in contrast to most of the previous works. In recent years Liu et al. (2018a) introduce a cross language attention model for event detection where they focus on English and Chinese. Liu et al. (2018b) propose a novel framework to jointly extract multiple event triggers and arguments. Sha et al. (2018) propose a novel dependency bridge RNN which includes syntactic dependency relationships. Dependency relationship is also used by Nguyen and Grishman (2018). They investigate a CNN based on dependency trees to perform event detection. Orr et al. (2018) present a Gated Recurrent Unit (GRU) based model that combines both temporal structure along with syntactic information through an attention mechanism. Event extraction task has also been addressed in specialized tracks dedicated in Text Analysis Conference (TAC). Event extraction in disaster domain in English language is 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, significant attempt to build event extraction system in Indian languages is lacking. In recent times, some of the works are reported in (SharmilaDevi et al., 2017; Sristy et al., 2017; Kuila and Sarkar, 2017; Singh et al., 2017). To the best of our knowledge, this is the first attempt to solve four important subtasks of event extraction *viz.* event trigger detection, event trigger classification, argument trigger detection and argument trigger classification simultaneously in a *multi-task, multi-lingual* setting.

3 Task Description and Contributions

In this paper, we propose a *multi-task*, *multi-lingual* trigger detection and classification method for Hindi and Bengali in Disaster related news data. For a given Hindi/Bengali sentence, we perform the following tasks simultaneously:

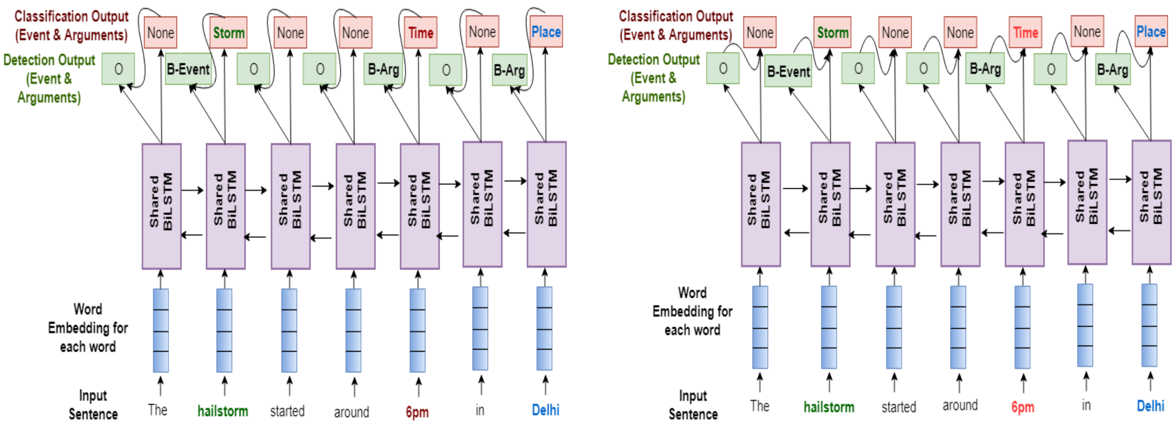
(a) Event Trigger Detection: Word or phrase that describes an event is called event trigger. Detecting event triggers is a sequence labeling task. But we formulate our current approach as a multi-class classification task as in (Chen et al., 2015; Ghaeini et al., 2016).

(b) Event Trigger Classification: Here the task is to classify each event trigger into predefined types.

(c) Argument Detection: Arguments are entities, times or values related to an event. Here the task is to detect such trigger words or phrase.

(d) Argument Classification: Classify each argument trigger into predefined argument roles.

Argument detection is also a sequence labeling task. Like event detection, we also formulate this task as a multi-class classification problem. In most of the previous works, both event and argument detection are considered as two separate tasks. However in our current work, we combine both the tasks into a single task based on our observation. Detailed analysis of news articles reveal the fact that each type of event triggers along with its corresponding arguments follow a particular pattern in a sentence. In the first example, the sentence contains *Place* argument **दिल्ली** (**Delhi**) and *Time* argument **शाम 6 बजे** (**6pm**). Each type of argument is followed by a type specific post-position ('में' for *Place* argument and 'के' for *Time* argument). In second example the sentence contains event specific argument like *Magnitude* (**7.2**) of earthquake along with *Place* argument **इंडोनिशया** (**Indonesia**). This type of patterns are often seen in news documents. So it is intuitive to consider



(a) Proposed System 1 (MT1)

(b) Proposed System 2 (MT2)

Figure 1: Architecture of Our Proposed Models

both event and argument trigger detection as a single task. For classification also, we merge both the event trigger classification and argument trigger classification as a single task. In this way, we learn all the four above mentioned tasks simultaneously using two loss functions. We perform our experiments using both Hindi and Bengali news datasets in mono-lingual as well as multi-lingual settings. We compare our multi-task learning (MTL) results with single-task learning (STL) results for the above mentioned mono-lingual and multi-lingual settings. For most of the cases we are getting 2% to 7% performance improvement in detection task. However for classification task, we see that the performance improves for some of the classes and for the remaining classes, the model does not perform at par with the other classes. Two contributions of our paper are

- A *multi-task*, *multi-lingual* approach for event extraction in Hindi and Bengali for disaster domain. Our proposed system has two variants - (a) The *classification* output helping in *detection* (MT1). (b) The *detection* output helping in *classification* (MT2). Both the architectures are discussed in methodology section.

- Provide a benchmark setup for event extraction in Hindi language.

The following examples show that each type of event and argument trigger is followed by semantically similar kind of words in a sentence. We highlight the **event trigger** and different types of argument triggers using different colour codes for better readability.

1. **Example-1 :** **दिल्ली** में **शाम 6 बजे** के आसपास **ओलावृष्टि** शुरू हुई।
   Transliteration : **dillee** mein **shaam 6 baje** ke aasapaas **olaavrshti** shuroo huee.
   Translation : The **hailstorm** started around **6pm** in **Delhi**.

2. **Example-2 :** **इंडोनिशया** में **7.2** की तीव्रता का **भूकंप** आया।
   Transliteration : **indoneshiya** mein **7.2** teevrata ka **bhookamp** aaya.
   Translation : There was a **7.2** magnitude **earthquake** in **Indonesia**.

3. **Example-3 :** **शुक्रवार** को **अफगानिस्तान** में हुई **विस्फोट** में **7 लोग** मारे गए हैं।
   Transliteration : **shukravaar** ko **aphagaanistaan** mein huee **visphot** mein **7 log** maare gae hain.
   Translation : **7 people** have been killed in an **explosion** in **Afghanistan** on **Friday**.

4 Methodology

Our proposed models take sentence of the form [w_0,w_1,...,w_n] as input. It produces two outputs for two main tasks namely *detection* (both event and argument) and *classification* (both event and argument). The *detection* task predicts the event or argument label (l_i) for each word (w_i) where $l_i \in$ I,O,B [1]. As we formulate *detection* as a multi-class classification task even though it being a sequence labeling task, we use *softmax* classifier at

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

	Hindi		Bengali		Multi-Lingual	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**
# of Document	681	194	799	199	1480	393
# of Sentences	12680	3077	20922	4635	33602	7712
# of Words	206882	50227	227234	45171	434116	95398
# of Event Triggers	5952	1533	7149	1602	13101	3135
# of Argument Triggers	36806	9244	44262	9058	81068	18302

Table 1: Dataset Statistics

the final layer. For *classification* task also, we use *softmax* classifier at the final layer to classify event and argument trigger into their predefined types. We employ a hard parameter sharing strategy (Caruana, 1993). We use a shared Bidirectional Long Short-Time Memory (Bi-LSTM) (Schuster and Paliwal, 1997) to capture the contextual information of each word. Figure 1a illustrates the design of first variant of our proposed architecture. Here the *classification* output of each word is concatenated with the corresponding representation resulting from the shared Bi-LSTM and fed as input to the final *detection* layer of that word. This is done with the intuition of improving the detection results with the help of classification output. For example if a word is classified as *'None'* then it has higher chance of being outside event or argument trigger boundaries. In subsequent sections, we call this architecture as MT1. Figure 1b illustrates the design of second variant of our proposed architecture. Here the *detection* output of each word is concatenated with the corresponding representation of the shared Bi-LSTM and fed as input to the final *classification* layer. This is done with the intuition of improving the classification results with the help of detection output.

4.1 Embedding

Each word of the input instance is converted to a numeric representation with the help of *fastText* (Grave et al., 2018) word embeddings having dimension 300 (d_e). The pre-trained word vectors are downloaded from *fastText* website[2]. To learn a mapping between mono-lingual word embeddings and obtain cross-lingual embeddings in order to bridge the language gap between two languages, we use the existing alignment matrices[3] which align monolingual vectors from two languages in a single vector space (Smith et al., 2017).

In order to handle *Out-of-Vocabulary* (OOV) words in the monolingual setting, we obtain their word embedding vectors from *fastText*'s `.bin` file. Separate vocabularies for OOV words are created for Hindi and Bengali respectively. We create separate `.vec` file for the two OOV vocabularies. We similarly transform these vectors of two different languages in a shared space using the existing alignment matrices[3]. It is seen that the performance has significantly improved using cross-lingual embeddings for OOV words compared to the method of using zero vectors for representing them.

5 Datasets and Experiments

5.1 Dataset

Words	Event & Argument Trigger Detection	Event & Argument Classification
इंडोनिशया	B_Arg	Place
में	O	None
7.2	B_Arg	Magnitude
की	O	None
तीव्रता	O	None
का	O	None
भूकंप	B_Event	Earthquake
आया	O	None

Table 2: Sample annotation for the sentence given in Example-2 in Task Description and Contribution Section

Since there is a lack of annotated data for our task, we create the datasets by crawling online Hindi and Bengali news articles and then annotate them following the TAC KBP[4] guidelines. For annotation, three annotators were employed. We estimate the inter-annotator agreement ratio by ask-

[2] https://fasttext.cc

[3] https://github.com/Babylonpartners/fastText_multilingual

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

ing all the three annotators to annotate 5% of total documents. The multi-rater Kappa (Fleiss, 1971) agreement ratio of 0.82 and 0.85 was observed for Hindi and Bengali news documents respectively.

For both the languages, news documents are crawled from online news portal. Every sentence of news documents was pre-processed for four sub-tasks of event extraction *viz.* event trigger detection, event trigger classification, argument detection and argument classification. Table 2 presents an example of sample annotation. For detection, we use IOB2[1] format (Ramshaw and Marcus, 1999). Our proposed Hindi dataset has two types of disaster events namely natural disaster and man-made disaster which are further classified into twenty seven sub-types. Each event trigger belongs to one of the twenty seven classes, which can be found in Table 8. Every event has multiple arguments of different roles. Hindi dataset contains eleven types of arguments excluding *Type* argument type. Bengali dataset also contains eleven type of arguments excluding argument type *Intensity*. Table 5 contains all the argument types. Some of the argument types common to both Hindi and Bengali, irrespective of the event types, are *Place*, *Time*, *Casualties* and *After-effect*. Some of the arguments are specific to some particular event types. For example, *Magnitude* and *Epicentre* are event specific arguments related to *Earthquake*. Table 1 presents the dataset statistics for training and the test set of Hindi and Bengali, respectively.

5.2 Experimental Setup

Epochs	300
# LSTM units	100
Loss function for Detection	categorical_crossentropy
Loss function for Classification	categorical_crossentropy
Optimizer	Adam

Table 3: Hyper-parameter Settings

For implementing the deep learning models Python based library Keras (Chollet et al., 2015) with Tensorflow (Abadi et al., 2015) backend is used. All the models are trained for 300 epochs. Training is done using a learning rate of 0.001 and *'Adam'* optimizer is used for fast convergence. The data is fed to the neural network in batches of 32. *'Checkpoints'* are used to save the best weights of the model based on training accuracy. Table 3 shows the hyper-parameter settings used in the implementation of both the variants of our proposed model. For evaluation *precision*, *recall* and *F1-score* are used as the metrics. However in result tables (refer Table 4, Table 5, Table 6, Table 7 and Table 8) only *F1-score* is reported.

6 Results and Analysis

Table 4, Table 5 and Table 8 show the experimental results for event and argument trigger detection, argument role classification and event trigger classification respectively, where ST denotes Single task, MT1 denotes Multi-task 1, MT2 denotes Multi-task 2 and SP denotes support count. Table 4 shows that multi-task model 1 (MT1) performs well as compared to single task (ST) model for all language settings. For each language setting, performance improvement is maximum in case of *I_Event* tag. We find that it is 7.3% for Hindi, for Bengali it is 11.5% and for multi-lingual setting it shows improvement of 6.5%. Analyzing the predictions of all the variants of our system reveal that words are usually miss-classified more between the Beginning (B) and Inside (I) tag type of either event or argument instead of events getting miss-classified as argument triggers. Thus we can conclude that the system produces near correct prediction of event and argument trigger in most of the cases, only issue being that it sometimes fail to determine the correct trigger boundary. Figure 2a and Figure 2b show the confusion matrix obtained by MT1 in trigger detection and trigger classification in the multilingual setting.

6.1 Comparison With Separate Event and Argument Trigger Detection System

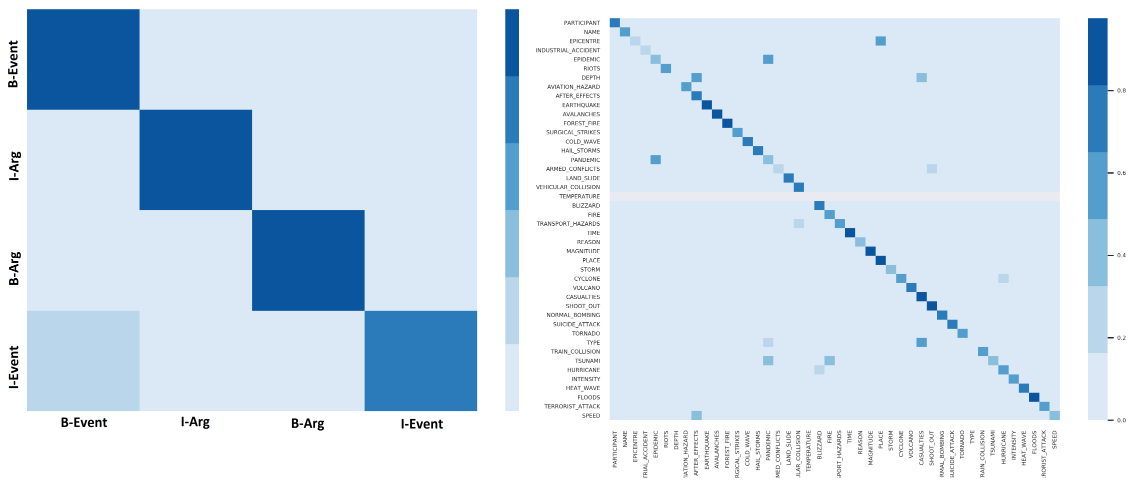
We also perform separate experiments to evaluate our proposed approach with the earlier proposed approaches of separately detecting event and argument triggers from sentences. Table 6 shows the *F1-score* achieved in event trigger detection and Table 7 shows the *F1-score* obtained in argument trigger detection for both the Hindi and Bengali datasets. The evaluation shows that there is not any significant loss in performance in simultaneous detection of event and argument triggers compared to individual trigger detection even though there is a marginal improvement in detection of the tag *I_Event* for Bengali in the argument detection model compared to the model which per-

	Hindi				Bengali				Multi-Lingual			
	ST	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**
B_Event	0.57	**0.59**	**0.59**	929	0.63	**0.65**	0.64	1111	**0.61**	**0.61**	**0.61**	2040
I_Event	0.41	**0.44**	0.42	594	0.52	**0.58**	0.55	491	0.46	0.48	**0.49**	1085
B_Arg	0.48	**0.5**	0.48	2476	0.57	0.57	**0.58**	2658	0.52	**0.53**	0.51	5134
I_Arg	**0.49**	**0.49**	0.46	6747	0.64	**0.65**	**0.65**	6400	0.56	**0.57**	0.55	13147

Table 4: Trigger Detection (Events and Arguments) Results

	Hindi				Bengali				Multi-Lingual			
	ST	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**
Participant	0.35	**0.42**	0.38	539	**0.43**	**0.43**	0.41	816	0.36	**0.41**	0.36	1355
Epicentre	**0.59**	0.46	0.29	22	**0.48**	0.27	0.46	49	**0.4**	0.2	0.35	71
After Effect	0.3	**0.35**	0.31	2828	**0.36**	**0.36**	0.35	1648	0.32	0.31	**0.33**	4476
Reason	**0.14**	0.1	0.12	354	**0.26**	0.21	0.20	280	0.16	0.16	**0.18**	634
Magnitude	0.56	0.6	**0.62**	40	**0.52**	0.51	0.44	25	0.47	**0.56**	0.54	65
Place	0.57	**0.58**	0.56	2369	**0.61**	0.59	**0.61**	1588	**0.58**	0.57	0.56	3957
Casualties	0.58	**0.59**	0.58	1969	**0.73**	**0.73**	0.72	2578	0.65	**0.66**	0.65	4547
Name	0.26	**0.32**	0.27	67	0	0	0	9	0.25	**0.3**	0.23	76
Type	-	-	-	-	0.20	0.20	**0.24**	29	0.19	0.11	**0.37**	29
Intensity	**0.54**	0.44	0.4	191	-	-	-	-	**0.45**	0.33	0.27	191
Time	0.65	**0.66**	0.63	804	0.84	**0.85**	0.84	2029	**0.79**	0.77	0.78	2833
Speed	**0.18**	0.11	0	17	0.36	0.31	**0.46**	4	0.19	**0.36**	0.27	21

Table 5: Argument Role Classification Results



(a) Trigger Detection (b) Trigger Classification

Figure 2: Confusion Matrix : MT1 in Multilingual Setting.

	Hindi	Bengali
B-Event	0.56	0.63
I-Event	0.42	0.55

Table 6: Result of Event Trigger Detection as Only Task.

	Hindi	Bengali
B-Arg	0.49	0.57
I-Arg	0.49	0.64

Table 7: Result of Argument Detection as Only Task.

	Hindi				Bengali				Multi-Lingual			
	ST	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**	**ST**	**MT1**	**MT2**	**SP**
Armed Conflicts	0.2	**0.4**	0.31	7	**0.22**	0.16	**0.22**	126	0.21	0.19	**0.24**	133
Avalanches	0.57	0.61	**0.62**	30	-	-	-	-	0.51	**0.57**	**0.57**	30
Aviation Hazard	0.35	0.43	**0.46**	43	**0.56**	0.47	0.34	34	**0.48**	0.34	0.41	77
Blizzard	0.49	**0.6**	0.51	19	0	0	0	7	0.44	0.41	**0.6**	26
Cold Wave	**0.53**	0.48	**0.53**	26	**0.50**	**0.50**	**0.50**	4	**0.52**	0.45	0.49	30
Cyclone	0.4	**0.49**	0.36	20	-	-	-	-	0.51	0.45	0.45	20
Earthquake	0.69	**0.73**	0.66	115	**0.75**	0.74	0.68	87	**0.71**	0.63	**0.71**	202
Epidemic	-	-	-	-	**0.33**	**0.33**	**0.33**	61	**0.34**	0.3	0.3	61
Fire	**0.27**	0.26	0.25	114	**0.68**	**0.68**	0.66	120	0.44	0.45	**0.48**	234
Floods	0.56	0.6	**0.7**	27	0.40	**0.67**	0.50	1	0.64	**0.77**	0.66	28
Forest Fire	**0.32**	0.31	0.29	63	-	-	-	-	**0.33**	0.3	0.24	63
Hail Storms	0.41	**0.46**	0.39	41	-	-	-	-	0.45	**0.52**	0.46	41
Heat Wave	0.39	**0.48**	0.39	66	0.33	0.24	**0.43**	9	0.36	0.37	**0.41**	75
Hurricane	0.53	**0.6**	0.38	35	-	-	-	-	**0.48**	0.47	0.45	35
Industrial Accident	**0.21**	**0.21**	0.17	113	0	**0.25**	0	3	0.17	**0.18**	0.15	116
Landslide	0.43	0.38	**0.44**	69	**0.74**	0.71	0.59	9	0.47	**0.5**	0.46	78
Normal Bombing	0.18	0.2	**0.22**	9	0.61	**0.62**	0.58	292	**0.57**	0.55	0.56	301
Pandemic	-	-	-	-	**0.26**	0.23	0.25	87	0.17	0.29	**0.32**	87
Riots	0.29	**0.38**	0.31	32	0.26	**0.31**	0.23	44	**0.28**	0.2	0.24	76
Shootout	**0.49**	**0.49**	0.44	110	**0.56**	0.54	0.52	177	0.51	**0.52**	0.5	287
Storm	0.2	0.22	**0.29**	24	**0.45**	0.42	0.42	26	**0.43**	0.32	0.34	50
Suicide Attack	0.64	0.64	**0.68**	154	0.57	**0.62**	0.56	123	**0.6**	0.59	0.58	277
Surgical Strikes	0	0	0	2	0.40	0.36	**0.44**	64	**0.41**	0.38	0.36	66
Terrorist Attack	0.61	0.61	**0.62**	95	0.32	**0.37**	0.34	147	0.47	0.48	**0.49**	242
Tornado	0.43	**0.49**	0.35	32	**0.57**	**0.4**	**0.57**	4	**0.43**	0.38	**0.43**	36
Train Collision	0.52	0.44	**0.53**	72	0	0	0	1	0.46	0.4	**0.5**	73
Transport Hazards	0.13	**0.18**	**0.18**	79	**0.49**	0.47	0.43	127	**0.4**	0.36	0.37	206
Tsunami	-	-	-	-	**0.17**	**0.17**	**0.17**	10	**0.32**	0.13	0.12	0.32
Vehicular Collision	**0.56**	0.52	0.49	93	0.43	0.45	**0.48**	39	0.44	**0.48**	0.46	132
Volcano	0.5	0.42	**0.52**	33	-	-	-	-	**0.48**	0.45	0.43	33

Table 8: Event Trigger Classification Results

forms simultaneous detection of both triggers.

6.2 Error Analysis

In the following Input Example 1, ज्वालामुखी विस्फोट (**jvaalaamukhee visphot\volcanic eruptions**) is a multi-word event trigger. The tags assigned for this trigger are *B_Event* and *I_Event* respectively. In Input Example 2, the event trigger विस्फोट (**visphot\eruptions**) is tagged as *B_Event*. For the first case, all the variants of the system predict the event trigger correctly but for the later case, our single task detection system (ST) and multi-task system 2 (MT2) predict it as outside event and argument trigger boundary (*O*) but multi-task system 1 (MT1) predicts it as inside event trigger (*I_Event*) rather than beginning of event trigger (*B_Event*). Thus we can see that all the variants miss-classify the trigger tag with MT1 being able to produce partially correct prediction as it, at least, classifies it to be of event type. However the classification result of the said event trigger in example 2 is correctly predicted by MT1 but it is wrongly predicted by MT2. Here we can see that the classification task is helping in detection task.

1. **Input Example 1 :** अमरीका में **ज्वालामुखी विस्फोट** को लेकर रेड अलर्ट जारी ।
   Transliteration : amareeka mein **jvaalaamukhee visphot** ko lekar red alart jaaree.
   Translation : US issues red alert for **volcanic eruptions**.

2. **Input Example 2 :** उल्लेखनीय है कि बीते कुछ दिनों से माउंट अगुंग ज्वालामुखी में छोटे-छोटे **विस्फोट** हो रहे है।
   Transliteration :ullekhaneey hai ki beete kuchh dinon se maunt agung jvaalaamukhee mein chhote-chhote **visphot** ho rahe hai.
   Translation : It is notable that in the last few days, small **eruptions** in the Mount Agung Volcano.

We provide below a detailed error analysis of the results achieved in classification task (refer to Table 5 and Table 8).

1. In the classification task (refer to Table 5), error analysis reveals that the performance is affected mainly due to two cases : (a) when the Support count of a trigger type is less, (b) when each trigger mention in a sentence is long, `i.e.` it consists of numerous words. For example, *Participant, Time, Place, Casualties* and *Intensity* have better *F1-score* as the trigger mentions corresponding to these types are in the form of short phrases as well as these types have larger support count. However, roles like *After Effect* and *Reason* have comparatively lower performance as these trigger mentions appear in sentences in the form of long phrases. Even though *Magnitude* has less support count, performance is better compared to the other roles as the trigger mention is in the form of a single word comprising of a numeric figure.

In Table 8, we observe the following drawbacks which can possibly lead to erroneous output.

1. We find that performance decreases for similar types of events. For example, types like *Fire, Forest Fire and Industrial Accident* are of similar type. We see that the performance of these types is low in Hindi as all of them are present in the dataset, thereby getting miss-classified. However in Bengali dataset, we find *Fire* performs relatively better as there does not exist any sentence having event trigger of type *Forest Fire* and *Industrial Accident*.

2. In Hindi dataset, we find that type *Transport Hazard* is seen to be misclassified with type *Train Collision* and type *Vehicular Collision*, therby leading to poor performance. For Bengali dataset, there hardly exists any trigger of type *Train Collision* and event trigger of type *Vehicular Collision* exists in small number. Thus Bengali dataset performs much better for *Transport Hazard*.

7 Conclusion and Future Works

In this paper, we present a *multi-tasking, multilingual* architecture for simultaneous *detection* and *classification* of event and argument triggers. We have proposed two variants where in each one of them, one task is helping another related task. Our results show that related tasks can definitely share information between them. We also compare our approach with separate models which can be employed for event and argument trigger detection respectively.

Other future works include developing an end-to-end system which will consist of a *multi-tasking* system such that given a sentence as input, event and argument triggers will be extracted from it and if there exists any link between the extracted event and argument, then the output of the system will be positive and otherwise negative.

8 Acknowledgement

The work reported in this paper is supported by the project titled “A Platform for Cross-lingual and Multi-lingual Event Monitoring in Indian Languages”, sponsored by IMPRINT-1, Ministry of

Human Resource and Development, Government 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.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Grégoire Burel, Hassan Saif, Miriam Fernandez, and Harith Alani. 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*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 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.

François Chollet et al. 2015. Keras. `https://keras.io`.

André Dittrich and Christian Lucas. 2014. Is this twitter event a disaster? *AGILE Digital Editions*.

Xiaocheng Feng, Bing Qin, and Ting Liu. 2018. A language-independent neural network for event detection. *Science China Information Sciences*, 61(9):092106.

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

Reza Ghaeini, Xiaoli Fern, Liang Huang, and Prasad Tadepalli. 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.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, pages 3483–3487.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 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.

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

Bernhard Klein, Federico Castanedo, Inigo Elejalde, Diego López-de Ipina, and Alejandro Prada Nespral. 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.

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

Qi Li, Heng Ji, and Liang Huang. 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.

Shasha Liao and Ralph Grishman. 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.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018a. Event detection via gated multilingual attention mechanism. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 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.

Shulin Liu, Yubo Chen, Kang Liu, Jun Zhao, et al. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 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.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 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.

Thien Huu Nguyen and Ralph Grishman. 2015. 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.

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolutional neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 886–891.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tim Nugent, Fabio Petroni, Natraj Raman, Lucas Carstens, and Jochen L Leidner. 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.

Walker Orr, Prasad Tadepalli, and Xiaoli Fern. 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.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Sebastian Riedel and Andrew McCallum. 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.

Sebastian Riedel and Andrew McCallum. 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.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

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

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

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

Jyoti Prakash Singh, Yogesh K Dwivedi, Nripendra P Rana, Abhinav Kumar, and Kawaljeet Kaur Kapoor. 2017. Event classification and location prediction from tweets during disasters. *Annals of Operations Research*, pages 1–21.

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Nagesh Bhattu Sristy, N Satya Krishna, and Durvasula VLN Somayajulu. 2017. Event extraction from social media text using conditional random fields. In *FIRE (Working Notes)*, pages 140–143.

Hristo Tanev, Jakub Piskorski, and Martin Atkinson. 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.

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 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.

Bishan Yang and Tom Mitchell. 2016. Joint extraction of events and entities within a document context. *arXiv preprint arXiv:1609.03632*.

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