Multi-lingual Event Identification in Disaster Domain

Zishan Ahmad, Deeksha Varshney, Asif Ekbal, Pushpak Bhattacharyya

Department of Computer Science and Engineering, Indian Institute of Technology Patna
{1821cs18,1821cs13,asif,pb}@iitp.ac.in

Abstract Information extraction in disaster domain is a critical task for effective disaster management. A high quality event detection system is the very first step towards this. Since disaster annotated data-sets are not available in Indian languages, we first create and annotate a dataset in three different languages, namely *Hindi*, *Bengali* and *English*. The data was crawled from the different news websites and annotated with expert annotators using a proper annotation guidelines. The events in the dataset belong to 35 different disaster classes. We then build a deep ensemble architecture based on Convolution Neural Network (CNN) and Bi-directional Long Short Term Memory (Bi-LSTM) network as the base learning models. This model is used to identify event words and phrases along with its class from the input sentence. Since our data is sparse, the model yields a very low F1-score in all the three languages. To mitigate the data sparsity problem we make use of multi-lingual word embedding so that joint training of all the languages could be done. To accommodate joint training we modify our model to contain language-specific layers so that the syntactic differences between the languages can be taken care of by these layers. By using multi-lingual embedding and training the whole dataset on our proposed model, the performance of event detection in each language improves significantly. We also report further analysis of language-wise and class-wise improvements of each language and event classes.

1 Introduction

Event is an occurrence that happens at a place and a particular time or time interval. Identification of events from textual data is not only an important task in information extraction but has a lot of practical applications as well. Automatically extracting events becomes an interesting task in information extraction because of the complexities involved in the text, and due to the fact that an event description may be spread over several other words and sentences. With the advent of internet, a huge amount of data is created each day in the form of news, blogs, social media posts, etc. and extracting relevant information from these is a huge challenge due to their built-in multi-source nature. The information extraction systems of today have to deal not only with isolated text, but also with large-scale and small-scale repositories occurring in different languages.

Event identification entails identifying the word or phrase that represents an event and also deciding the type of event from a set of pre-defined labels. Automatic event extraction system based on machine learning requires a huge amount of data. Collecting and annotating such data is both time consuming and expensive. Creating such a dataset for multiple languages becomes even more challenging, because finding good annotators for diverse languages is also difficult and expensive. In this paper we create and publish a dataset annotated for events in disaster domain in three different languages, namely *Hindi*, *Bengali* and *English*. We then build a deep learning model, based on CNN (Convolutional Neural Network) and Bi-LSTM (Bi-Directional Long Short Term Memory) for the task of event identification.

Since the dataset is small for a particular language, we leverage the information of the other languages while training. In order to achieve this we make use of multi-lingual word embeddings to bring all the language representations to a shared vector space. We show that by training our model in such a way we are able to utilize the dataset of all the three languages and improve the performance of our system for each language. We also build a second model using deep learning with separate language-specific layers. By using this model along with multi-lingual word embedding, we are able to further improve the performance for all the languages. Our empirical results show that, for the task of event identification, sharing of knowledge across the languages helps in improving the overall performance of the system.

1.1 Problem Definition

Event Identification or detection is a sequence labelling task. Given a sentence in *Hindi*, *Bengali* or *English* of the form $w_1, w_2, w_3, ...w_n$ the task is to predict the best label sequence of the form $l_1, l_2, l_3, ...l_n$. In order to properly denote the boundaries, a multi-word event trigger is encoded in terms of IOB [26] notation, where B, I and O denote the beginning, intermediate and outside token of an event. There are 35 different disaster class labels which need to be identified. The example mentioned below depicts the input sentence and output label sequence. Event triggers are boldfaced in the example.

- **Input Hindi Sentence:** गृह मंत्रालय मुंबई के बम विस्फोटों के मद्देनजर इस बात की विशेष तौर पर जांच कर रहा है कि अक्षरधाम मंदिर और १९९३ के मुंबई बम विस्फोटों के फैसलों की प्रतिक्रिया के रूप में तो यह हमले नहीं हुए
- **Transliteration**: grih mantraalay mumbai ke **bam visphoton** ke maddenajar is baat kee vishesh taur par jaanch kar raha hai ki aksharadhaam mandir aur 1993 ke mumbai **bam visphoton** ke phaisalon kee pratikriya ke roop mein to yah **hamale** nahin hue
- **Translation**: In view of the Mumbai **bomb blasts**, the Home Ministry is specially investigating the fact that these **attacks** did not take place as response to the Akshardham Temple and the 1993 Bombay **bomb blasts**
- **Output**: O O O O I_Terrorist_Attack I_Terrorist_Attack O O O O O O O O O O O O O O O O O O O O I_Terrorist_Attack I_Terrorist_Attack O O O O O O O O O O I_Terrorist_Attack O O

2 Related Work

Event extraction is a well-researched problem in the area of Information Extraction, particularly for event detection and classification. Some of the well-known works can be found in [22, 4, 6].

The early approaches were based on pattern matching in which, patterns were created from predicates, event triggers and constraints on its syntactic context [8, 3, 2, 29]. Later, feature based methods were used to learn a better representation for sentences by forming rich feature sets for event detection models ranging from lower-level representations [13, 30, 7] to higher-level representations such as cross-sentence or cross-event information [11, 15, 16, 10, 14].

The major disadvantages associated with the traditional feature based methods are due to the complexity associated with handcrafted feature engineering. To overcome this, Nguyen and Grishman (2015) [22] used CNN to automatically extract efficient feature representations from the pre-trained embeddings (word, position and entity-type) for event extraction. For multi-event sentences, Chen et al. [4] introduced a dynamic multi-pooling layer according to event triggers and arguments into CNN to capture more crucial information.

In 2016, Nguyen and Grishman [23] proposed a technique that made use of non-consecutive convolution for sentences which skip unnecessary words in word sequences. Feng et al. [6] combined the representations learned from a CNN with a Bidirectional LSTM [9] to learn the continuous representations of a word. Further in the same year, Nguyen and others [21] use a variant of LSTM called the Gated Recurrent Units (GRU) [5] in association with a memory network to jointly predict events and their arguments. Often event detection suffers from data sparseness. To handle this Liu et al. [19] used FrameNet to detect events and mapped the frames to event-types, thus obtaining extra training data. Liu et al. [20] further extended the work by building an extraction model using the information from arguments and FrameNet using supervised attention mechanism.

Recently, neural network models involving dependency trees have also been attempted. Nguyen and Grishman (2018) [24] explored syntactic representations of sentences and examined a convolutional neural network based on dependency trees to perform event detection. Sha et al. [27] came up with dependency bridges over Bi-LSTM for event extraction. Their work was further extended by Orr et al. [25], who used attention to combine syntactic and temporal information. Liu et al. [18] built a Gated Multi-Lingual Attention (GMLATT) framework which used monolingual context attention to gather information in each language and gated cross-lingual attention to combine information of different languages. They produced their results for English and Chinese languages. Feng et al. [6] illustrated the techniques for building a language independent event extraction system. Lin et al. [17] proposed a multi-lingual system for sequence labelling. They used closely related languages and used character CNNs to get the representation of words in languages that share alphabets.

In this paper, instead of using character CNNs to get shared representation of a word, we use multi-lingual embeddings. This gives us the freedom to do joint

learning between the languages that do not share the same characters. We also demonstrate that our method can be used for event identification for languages that are diverse and syntactically dissimilar like *English*, *Hindi* and *Bengali*.

3 Methodology

We develop three models based on deep learning for conducting our experiments. First we build a deep learning model based on Bi-Directional Long Short Term Memory (Bi-LSTM)[9] and Convolution Neural Network (CNN)[12]. We train the model separately for each of the three languages using monolingual word embedding. We then leverage the information from the other languages, by following two approaches: i). We use the same deep learning model to train with all the languages simultaneously with multi-lingual word embedding as features; ii). In the second model separate MLP (Multi Layer Perceptron) is used for each language at the final layer. Here too the training is done with all the languages at the same time using multi-lingual word embedding as features.

3.1 Mono-lingual Word Embedding Representation

The pre-trained monolingual word-embeddings used in the experiments are popularly known as fastText[1][1]. This method makes use of, continuous skip-gram model and is used to obtain the word-embedding representation of size 300 for each word. The advantage of using fastText is that even if some words are not available in the training corpus, their representations can still be obtained. Thus a better representation of out-of-vocabulary words are obtained by using fastText. The skip-gram model of fastText is trained on *Wikipedia* data dumps of their respective language.

3.2 Multi-lingual Word Embedding Representation

We use alignment matrices[2] to align the embedding of different languages ($L1$,$L2$ and $L3$) to a single vector space. The word embedding representations obtained from fastText for language $L1$ are multiplied with the alignment matrices $L1$ to transform the word embedding to a shared vector space. This method of transforming the word embedding was proposed in Smith et al[28]. They proved that a self-consistent linear mapping between the semantic spaces must be orthogonal. Two parallel dictionaries between languages, D_{L2-L1} and D_{L3-L1} are used for this purpose. By aligning all the three-language embedding to the vector space of $L1$, we obtain multi-lingual word embedding. The following Equation 1 is used to obtain the alignment matrix.

$$\max_{O} \sum_{i=1}^{n} y_i^T O x_i, \quad \text{Subject to } O^T O = I \tag{1}$$

[1] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md
[2] https://github.com/Babylonpartners/fastText_multilingual

Where x_i and y_i belong to a dictionary $\{x_i, y_i\}_1^n$ of paired words of two languages $L1$ and $L2$ (or $L1$ and $L3$) respectively, and O is the orthogonal transformation matrix. Single Valued Decomposition (SVD) is then used to accomplish the objective of Equation 1. Matrices U and V are obtained, such that $O = UV^T$. By applying the transformation V^T to the source language $L1$ and U^T to the target language $L2$, both the languages can be mapped into a single vector space. The same thing is repeated for $L1$ and $L3$. For our experiments we transform *Hindi* and *Bengali* language word embeddings to the vector space of *English* word embeddings.

3.3 Baseline Model for Event Identification

The event identification model used for mono-lingual setting is shown in Figure 1. The task of event identification is formulated as a sequence labeling problem. For each input token we need to decide if the token belongs to an event and also what event type it belongs to. The input to the model is a sentence, represented by a sequence of word embeddings. Since Bi-LSTM takes fixed input sizes the smaller sentences are padded with zero vectors and brought to equal length. This sequence is passed to Bi-LSTM and CNN of filter sizes 2 and 3. The Bi-LSTM yields output representation of each word and CNN gives convoluted bi-gram and tri-gram features. These features are concatenated and passed through a Multi Layer Perceptron (MLP), followed by a Softmax classifier which gives the probability distribution over the possible tags I_Event_Type or O_Event.

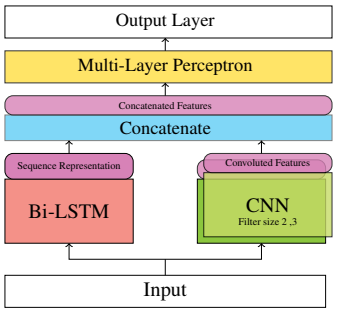


Figure 1: Architecture of baseline model

3.4 Proposed Multilingual Event Identification Model

We develop a model similar to the baseline model for multilingual event identification. The input sentence to this model is represented by sequence of multilingual word-embeddings. We create separate MLP layer for each language, such that for an input data of language *L1*, only the MLP of language *L1* will be used. The backpropagation will take place only through MLP of language *L1*, and the weights of other language MLPs will not be updated. All the layers before MLPs will be updated for every input, irrespective of the languages. Thus,

the Bi-LSTM and CNNs produce a shared representation that is used by all the MLPs. The separate MLPs act as language specific decoders which decodes event representations of each language from the shared representation of languages.

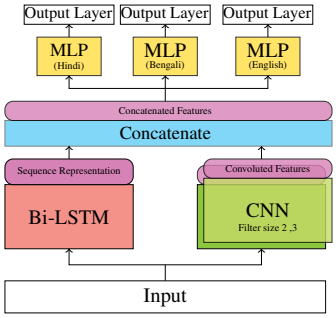


Figure 2: Architecture of the proposed multilingual event identification model

4 Datasets and Experiments

In this section we provide the details of the datasets used in our experiments along with the experimental setup.

4.1 Datasets

We use a multi-lingual setup, comprising of three languages, namely English, Hindi and Bengali. The dataset was created by crawling the popular news websites in the respective languages. All the news crawled are from the disaster events (both man-made and natural). All the news documents were annotated by three annotators, with good linguistic knowledge and having sufficient knowledge of the related area. The annotation guideline followed was similar to the annotation guideline provided by TAC KBP 2017 Event Sequence Annotation Guidelines[3]. The multi-rater Kappa agreement ratio of the annotators was found to be 0.85 on an average.

The total dataset is comprising of 2,191 documents (*Hindi*: 922, *Bengali*: 999 and *English*: 270). It comprises of 44,615 sentences (*Hindi*: 17,116, *Bengali*: 25,717 and *English*:1,782) and a total of 596,423 words (*Hindi*: 276,155, *Bengali*: 273,115 and *English*: 47,153). Thirty five different disaster types were chosen as class labels for annotation. The classes and the distribution of classes in the three language datasets are detailed in Table 1.

4.2 Experimental Setup

For implementing the deep learning model and conducting the experiments, a Python based library Keras[4] was used. Since both the models are based on Bi-

[3] https://cairo.lti.cs.cmu.edu/kbp/2017/event/TAC_KBP_2017_Event_Coreference_and_Sequence_Annotation_Guidelines_v1.1.pdf

[4] http://keras.io

Table 1: Event types in the dataset, and number of triggers for each class in *Hindi*, *Bengali* and *English* datasets

Class	Hindi	Bengali	English	Class	Hindi	Bengali	English
Epidemic	0	256	29	*Forest Fire*	327	10	34
Hurricane	51	25	23	*Terrorist Attack*	507	741	31
Drought	5	8	6	*Vehicular Collision*	482	436	91
Earthquake	145	325	177	*Industrial Accident*	329	70	35
Shootout	545	942	194	*Volcano*	238	4	91
Blizzard	140	13	9	*Land Slide*	225	33	30
Surgical Strikes	2	344	34	*Train Collision*	325	32	53
Fire	470	600	154	*Pandemic*	0	513	0
Tsunami	10	56	11	*Riots*	311	148	14
Storm	601	154	79	*Armed Conflicts*	44	399	6
Normal Bombing	89	1783	40	*Cyclone*	148	21	47
Avalanches	130	1	23	*Seismic Risk*	0	1	0
Suicide Attack	750	613	15	*Transport Hazards*	379	882	94
Tornado	159	12	39	*Aviation Hazard*	174	238	41
Floods	231	43	55	*Cold Wave*	134	19	39
Heat Wave	438	65	11	*Hail Storms*	184	0	32
Limnic Eruptions	0	0	1	*Famine*	0	0	1
Rock Fall	0	0	6				

LSTM the input sequence lengths needed to be the same. Padding by zero vector of length 300 was used to make all the sequences equal in length. The sequence length was fixed to 75. *Relu* was used for activation and *Dropout* of 0.3 was used for all the intermediate layers in the model. For the baseline event identification model, one MLP with two linear layers were used. The final layer consists of 35 neurons (since there are 35 classes), and *Softmax* is used for classification of the final output. For the multi-lingual event identification model, three MLPs are used in parallel for the three languages. Each MLP contains two linear layers, with final layer containing 35 neurons. *Softmax* is used for the classification of final output from all the three MLPs.

5 Results and Analysis

We present all the experimental results in this section. The first experiment (*Exp-1*) is conducted on our baseline model (c.f Figure 1), by using monolingual word embedding as the input features. This model is trained separately for each language, thus we have individual models for each of the three languages.

Table 2: Macro-averaged *Precision (P)*, *Recall (R)* and *F1-score (F)* for the three experiments: 5-fold cross-validated

	Exp-1			Exp-2			Exp-3		
Classes	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
Hindi	0.32	0.25	0.25	0.32	0.25	0.26	0.40	0.37	**0.36**
Bengali	0.22	0.18	0.18	0.33	0.25	0.26	0.35	0.29	**0.30**
English	0.18	0.21	0.18	0.33	0.29	0.28	0.43	0.38	**0.39**

In the second experiment (*Exp-2*) multi-lingual word embeddings are used as the input features. The datasets of all the languages are shuffled and taken

together for training. Thus in the second experiment we have only one model that operates for all the three languages. The third experiment (*Exp-3*) is conducted on the multilingual event identification model (c.f Figure 2). In this experiment we use multi-lingual word embeddings as input feature, and the training is done with all the languages together.

The results of these experiments are shown in Table 3. From the results it can clearly be seen that the performance of all the languages improve in our second setting. The improvement for the *Hindi* is the least while that of *English* is the most. The results clearly show that using multi-lingual embeddings and all the three-language datasets for training helps in training and produces better results, than training individual models using mono-lingual word embeddings. After conducting the third experiment (i.e. Exp-3) the results improve even further for every language. This shows that a separate language layer for each language helps for better learning in the multi-lingual setting. Separate layers act as the special language decoders that take shared representation from the previous layers and learn the best mapping from this to the language output. The best jump in F1-score is seen in *English*. This is because the *English* dataset is the smallest in size, and thus the the baseline model for this dataset is under-trained. The F1-score of *English* dataset more than doubles (from 0.18 to 0.39) when using multi-lingual settings and separate language layers. The F1-score for *Hindi* and *Bengali* datasets improve by 11% and 12% respectively, over the baseline.

The class-wise F1-score for each of the three experiments for all the datasets are shown in Table 2. For each class the best F1-score is shown in bold. It can be clearly seen that the third setup outperforms the first and second setups for most of the classes for all the three languages. We observe the most performance improvement for the *English* language. For many classes the initial model could not predict any instance while the final model was able to give a good F1-score. This usually happens when one of the language datasets has a good number of instances for that particular class. For e.g. the class *Blizzard* has 140 instances in *Hindi* language dataset, while only 9 instances are present in the *Bengali* and *English* datasets. The initial prediction of the experiment yields an F1-score of 0 for *English*, while the third experiment boosts the F1-score to 0.88. This clearly shows that sharing of knowledge across the different languages do help each other.For class like *Train Collision* 352 instances are present in *Hindi* dataset while only 32 and 53 instances are present in *Bengali* and *English* dataset, respectively. In this case we see that the system could not predict the class for *Bengali* and *English* dataset for *Exp 1* and *Exp 2*. However by using separate layers the system was able to predict the class *Train collision*. This shows the effectiveness of separate layers even when only one of the datasets has enough instances of a class and others do not. Few class instances like *Limnic Eruptions*, *Rock Fall* and *Famine* are only present in the *English* dataset with a few instances.For such classes we see no improvement in performance. Thus improvement is only noticed when enough number of instances are present in at least one of the languages.

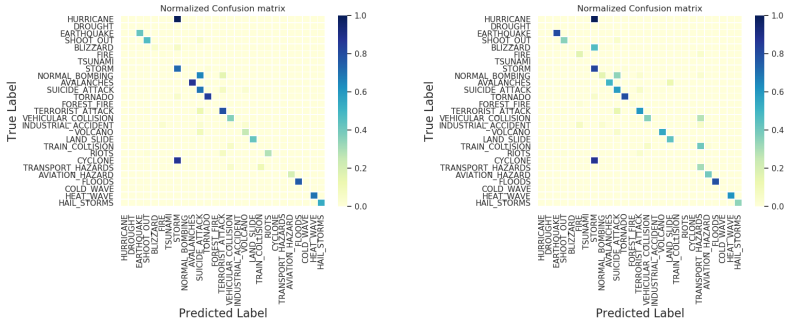
Table 3: *F1-score* for the three experiments on *Hindi (H)*, *Bengali (B)* and *English (E)* datasets: 5-fold cross-validated ('-' represents the absence of instances of the class for that language)

	Exp-1			**Exp-2**			**Exp-3**		
Language	**H**	**B**	**E**	**H**	**B**	**E**	**H**	**B**	**E**
Drought	0	0	0	0.32	0	0	**0.67**	**0.06**	0
Earthquake	0.38	0.80	0.82	**0.53**	0.82	0.93	0.49	**0.83**	**0.96**
Shoot out	0.57	0.65	0.58	**0.58**	**0.65**	0.65	0.55	0.58	**0.67**
Blizzard	**0.57**	0	0	0.17	0	00	0.53	0	**0.88**
Fire	0.12	0.70	0.58	0.2	**0.73**	0.67	**0.29**	0.69	**0.68**
Tsunami	0	0	0	0	0	0	**0.41**	**0.37**	**0.13**
Storm	0.54	**0.66**	0.55	0.48	0.17	0.51	**0.54**	0.4	**0.72**
Normal Bombing	0	0.60	**0.39**	0.17	**0.72**	0.37	**0.22**	0.68	0.32
Avalanches	0.67	0	0.28	0.67	0	0.65	**0.78**	0	**0.87**
Suicide Attack	0.67	0.52	0	0.68	0.63	0.28	**0.68**	**0.68**	**0.54**
Tornado	0.43	0	0.59	0.52	0	0.95	**0.52**	**0.33**	**0.99**
Forest Fire	0.14	0	0	0.04	0	0	**0.38**	0	**0.32**
Terrorist Attack	**0.68**	0.39	0	0.67	**0.46**	0.21	0.66	0.43	**0.29**
Vehicular Collision	0.45	0.33	0.22	0.42	0.44	0.38	**0.5**	**0.45**	**0.52**
Industrial Accident	0.02	0	0	0	0	**0.30**	**0.20**	0	0
Volcano	0.41	0	0.67	0.41	0	0.63	**0.50**	0	**0.69**
Land Slide	0.64	0	0.13	**0.67**	0.36	0.65	0.65	**0.48**	**0.8**
Train Collision	0.14	0	0	0.06	0	0	**0.38**	**0.12**	**0.17**
Riots	0.14	0	0	0.10	**0.24**	**0.10**	**0.36**	0.23	0.07
Armed Conflict	0	0.37	0	0.03	**0.47**	**0.29**	**0.17**	0.44	0
Cyclone	0.03	0	0.63	0	0	0.37	**0.29**	0	**0.80**
Transport Hazard	0.07	0.53	0.16	0.11	**0.56**	0.25	**0.24**	0.53	**0.40**
Aviation Hazard	0.14	0.08	0	**0.48**	**0.64**	**0.52**	0.42	0.63	0.45
Floods	0.62	0	0.49	**0.62**	**0.65**	0.64	0.60	0.56	**0.69**
Cold Wave	0.24	0	0.24	0.20	0.22	0.72	**0.56**	**0.40**	**0.90**
Heat Wave	0.59	0	0	**0.59**	**0.38**	0.09	0.53	0.28	**0.5**
Hail Storm	0.63	0.64	0.07	0.62	0	0.16	**0.66**	**0.66**	**0.28**
Hurricane	0	0	0	0	0	0.12	**0.15**	**0.09**	**0.79**
Surgical Strike	0	0.02	0	0	0.46	0.07	0	**0.50**	**0.27**
Epidemic	-	0.1	0	-	0.07	0	-	**0.40**	0
Pandemic	-	0.63	**0.63**	-	**0.64**	0	-	0.56	0.56
Seismic Risk	-	0	0	-	0	0	-	0	0
Limnic Eruptions	-	-	0	-	-	0	-	-	0
Rock Fall	-	-	0	-	-	0	-	-	0

5.1 Error Analysis

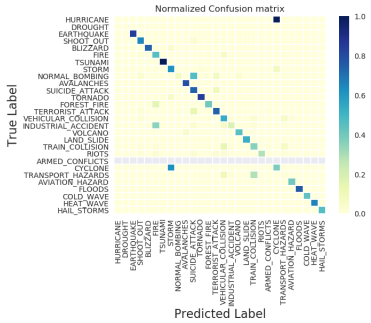
We provide analysis of the errors produced by our models in this section. Since the event identification problem classifies the words or phrases into different classes, we use confusion matrices[5] to determine which classes the model is confused with. We observe that miss-classifications reduce for all the languages from *Exp 1* to *Exp 3*. From the confusion matrix of *Hindi*, shown in Figure 3a, it can be clearly seen that the system was getting confused between the types that were very close to each other. For example *Blizzard* was getting confused with *Storm*. In this situation multilingual event identification model showed significant improvement (c.f Figure 3c). It was able to identify accurately nearly 60% instances of *Blizzard* in the *Hindi* dataset. Even though the instances of the class *Blizzard* are very few in other languages, the instances of *Storm* are plenty in

[5] The confusion matrices for each language dataset were computed on 80:20 train-test split



(a) Confusion matrix Exp-1

(b) Confusion matrix Exp-2

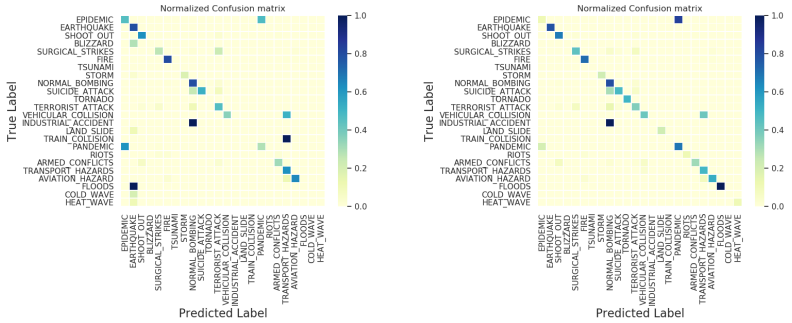


(c) Confusion matrix Exp-3

Figure 3: Confusion matrix of Exp-1, Exp-2 and Exp-3 for *Hindi* language test set

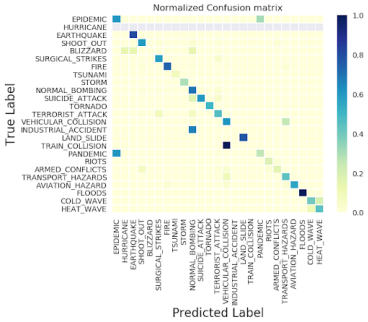
the other datasets. This helps the model in creating a representation that can better discriminate between the classes *Storm* and *Blizzard*- thus improving the performance for both the classes. In Figure 3a, we see that the model was almost unable to identify the classes *Fire* and *Tsunami*. However, using the multilingual model it could identify these classes better. This might be attributed because for both of these events, *Bengali* dataset helps by providing many instances of these classes (c.f Figure 3c). Similarly, good presence of the events *Landslide* and *Heatwave* in the *Hindi* dataset help in extracting these events in the *Bengali* dataset (c.f Figure 4c).

The baseline system for the *English* dataset was confused between the classes *Vehicular Collision*, *Train Collision*, *Transport Hazard* and *Aviation Hazard* (c.f Figure 5a). The source of confusion between these classes was the lack of enough data for these classes in the *English* dataset. However, these classes are very well covered by *Hindi* and *Bengali* datasets as seen in Table 1. This helps in greatly reducing the confusion between these classes (c.f Figure 5c) and improving the F1-scores of all these classes simultaneously. Furthermore, we observe that for all the languages some missing classes are predicted accurately in our final model. These classes are shown as gray colored rows in Figures 3c, 4c and 5c.



(a) Confusion matrix Exp-1

(b) Confusion matrix Exp-2



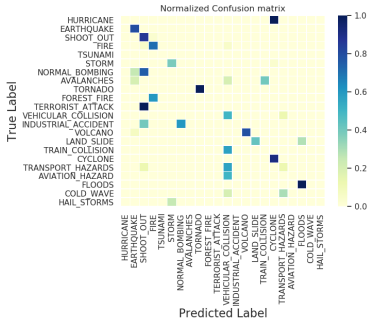
(c) Confusion matrix Exp-3

Figure 4: Confusion matrix of Exp-1, Exp-2 and Exp-3 for *Bengali* language test set

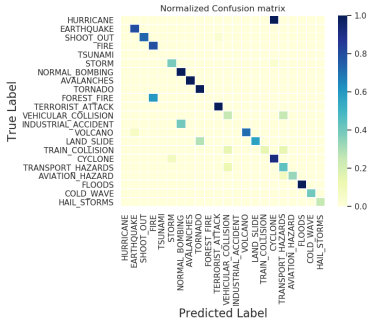
6 Conclusion and Future Works

In this paper we have proposed a deep neural network architecture for multi-lingual event extraction in disaster domain. We have created the event annotated datasets in three different languages, namely *Hindi*, *Bengali* and *English*. We build two different deep learning based models to identify events from a given text. We have used both mono-lingual and multi-lingual word embeddings as input features to our model. We have empirically shown that using multi-lingual word embeddings, we can leverage the information across the different languages using joint training, that eventually improves the performance of each language. We also show that using separate MLPs for each language further improves the performance of the system for all the languages.

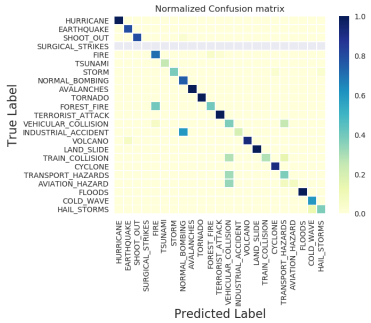
We make use of syntactically different languages like *English* and *Hindi* in our experiments, thus we can say that using multi-lingual word embeddings and language specific MLPs, the syntactic differences between languages can be handled, while also improving the performance of the system by joint training. This work can be used as a benchmark setup for future experiments on Event Identification in *Hindi*, *English* and *Bengali*. As future work, it would be interesting



(a) Confusion matrix Exp-1



(b) Confusion matrix Exp-2



(c) Confusion matrix Exp-3

Figure 5: Confusion matrix of Exp-1, Exp-2 and Exp-3 for *English* language test set

to add more languages of diverse nature to the experiments mentioned in this paper and observe the results.

7 Acknowledgement

The research reported in this paper is an outcome of the project titled "A Platform for Cross-lingual and Multi-lingual Event Monitoring in Indian Languages", supported by IMPRINT-1, MHRD, Govt. of India, and MeiTY, Govt. of India. The *Bengali* and *English* language datasets used in the experiments, were created by the project partners at Indian Institute of Technology Kharagpur, and Anna University - K. B. Chandrashekar Research Centre respectively.

References

1. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)

2. Cao, K., Li, X., Fan, M., Grishman, R.: Improving event detection with active learning. In: Proceedings of the International Conference Recent Advances in Natural Language Processing. pp. 72–77 (2015)
3. Cao, K., Li, X., Grishman, R.: Improving event detection with dependency regularization. In: Proceedings of the International Conference Recent Advances in Natural Language Processing. pp. 78–83 (2015)
4. Chen, Y., Xu, L., Liu, K., Zeng, D., Zhao, J.: 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). vol. 1, pp. 167–176 (2015)
5. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
6. Feng, X., Qin, B., Liu, T.: A language-independent neural network for event detection. Science China Information Sciences **61**(9), 092106 (2018)
7. Ferguson, J., Lockard, C., Weld, D., Hajishirzi, H.: Semi-supervised event extraction with paraphrase clusters. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). vol. 2, pp. 359–364 (2018)
8. Grishman, R., Westbrook, D., Meyers, A.: Nyu's english ace 2005 system description. ACE **5** (2005)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Hong, Y., Zhang, J., Ma, B., Yao, J., Zhou, G., Zhu, Q.: 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. pp. 1127–1136. Association for Computational Linguistics (2011)
11. Ji, H., Grishman, R.: Refining event extraction through cross-document inference. Proceedings of ACL-08: HLT pp. 254–262 (2008)
12. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
13. Li, Q., Ji, H., Huang, L.: 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). vol. 1, pp. 73–82 (2013)
14. Li, X., Nguyen, T.H., Cao, K., Grishman, R.: Improving event detection with abstract meaning representation. In: Proceedings of the First Workshop on Computing News Storylines. pp. 11–15 (2015)
15. Liao, S., Grishman, R.: Using document level cross-event inference to improve event extraction. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. pp. 789–797. Association for Computational Linguistics (2010)
16. Liao, S., Grishman, R.: Acquiring topic features to improve event extraction: in preselected and balanced collections. In: Proceedings of the International Conference Recent Advances in Natural Language Processing 2011. pp. 9–16 (2011)
17. Lin, Y., Yang, S., Stoyanov, V., Ji, H.: A multi-lingual multi-task architecture for low-resource sequence labeling. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 799–809 (2018)
18. Liu, J., Chen, Y., Liu, K., Zhao, J.: Event detection via gated multilingual attention mechanism. Statistics **1000**, 1250 (2018)

19. Liu, S., Chen, Y., He, S., Liu, K., Zhao, J.: Leveraging framenet to improve automatic event detection. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 2134–2143 (2016)
20. Liu, S., Chen, Y., Liu, K., Zhao, J.: Exploiting argument information to improve event detection via supervised attention mechanisms. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 1789–1798 (2017)
21. Nguyen, T.H., Cho, K., Grishman, R.: 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. pp. 300–309 (2016)
22. Nguyen, T.H., Grishman, R.: 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). vol. 2, pp. 365–371 (2015)
23. Nguyen, T.H., Grishman, R.: Modeling skip-grams for event detection with convolutional neural networks. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 886–891 (2016)
24. Nguyen, T.H., Grishman, R.: Graph convolutional networks with argument-aware pooling for event detection (2018)
25. Orr, J.W., Tadepalli, P., Fern, X.: Event detection with neural networks: A rigorous empirical evaluation. arXiv preprint arXiv:1808.08504 (2018)
26. Ramshaw, L.A., Marcus, M.P.: Text chunking using transformation-based learning. In: Natural language processing using very large corpora, pp. 157–176. Springer (1999)
27. Sha, L., Qian, F., Chang, B., Sui, Z.: Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
28. Smith, S.L., Turban, D.H., Hamblin, S., Hammerla, N.Y.: Offline bilingual word vectors, orthogonal transformations and the inverted softmax. arXiv preprint arXiv:1702.03859 (2017)
29. Tanev, H., Piskorski, J., Atkinson, M.: Real-time news event extraction for global crisis monitoring. In: International Conference on Application of Natural Language to Information Systems. pp. 207–218. Springer (2008)
30. Yang, B., Mitchell, T.: Joint extraction of events and entities within a document context. arXiv preprint arXiv:1609.03632 (2016)