# A Language Invariant Neural Method for TimeML Event Detection

**Suhan Prabhu, Pranav Goel, Alok Debnath** and **Manish Shrivastava**
International Institute of Information Technology
Hyderabad, Telangana, India
{suhan.prabhuk, pranav.goel, alok.debnath }@research.iiit.ac.in
m.shrivastava@iiit.ac.in

## Abstract

Detection of TimeML events in text have traditionally been done on corpora such as TimeBanks. However, deep learning methods have not been applied to these corpora, because these datasets seldom contain more than 10,000 event mentions. Traditional architectures revolve around highly feature engineered, language specific statistical models.

In this paper, we present a Language Invariant Neural Event Detection (ALINED) architecture. ALINED uses an aggregation of both sub-word level features as well as lexical and structural information. This is achieved by combining convolution over character embeddings, with recurrent layers over contextual word embeddings. We find that our model extracts relevant features for event span identification without relying on language specific features.

We compare the performance of our language invariant model to the current state-of-the-art in English, Spanish, Italian and French. We outperform the F1-score of the state of the art in English by 1.65 points. We achieve F1-scores of 84.96, 80.87 and 74.81 on Spanish, Italian and French respectively which is comparable to the current states of the art for these languages. We also introduce the automatic annotation of events in Hindi, a low resource language, with an F1-Score of 77.13.

## 1 Introduction

Automatic extraction of events has gained sizable attention in subfields of NLP and information retrieval such as automatic summarization, question answering and knowledge graph embeddings (Chieu and Lee, 2004; Glavaš and Šnajder, 2014), as events are a representation of temporal information and sequences in text. Various developments in guidelines and datasets for event detection have been met with equally fast paced evolution of automatic event annotation and detection methodologies in the last few years (Doddington et al., 2004; Pustejovsky et al., 2010; O'Gorman et al., 2016). On a larger scale, event extraction has extended to many languages beyond English, including French (Bittar et al., 2011), Spanish (Saurı, 2010), Italian (Caselli et al., 2011a) and very recently, Hindi (Goud et al., 2019b). Event detection architectures have their origins in statistical models such as K-means and hierarchical clustering methods (Arnulphy et al., 2015), which have more recently given way to neural models. Deep neural architectures on event annotation vary based on the approach taken to identifying and handling the data.

However, event detection as a problem shifts when we move away from the annotation paradigm of datasets such as ACE (Doddington et al., 2004) and TAC KBP (Mitamura et al., 2015) to TimeML datasets such as TimeBank (Pustejovsky et al., 2006), which are used in this paper. There has been limited use of deep learning methods on TimeBanks due to fewer event mentions and a need for data augmentation and bootstrapping. However, in this paper, we show that using subword level information, a language invariant deep learning model can provide similar event detection accuracies as heavily feature engineered language specific statistical methods without using any augmented data.

This paper has two main contributions. First, we introduce our model, the Architecture for Language Invariant Neural Event Detection (ALINED), which is a deep learning model for event extraction from TimeML event annotated datasets from five languages. We show that for four of these languages, using no augmented data, we achieve comparable F1 score on these datasets to heavily feature engineered language specific



*D M Sharma, P Bhattacharyya and R Sangal. Proc. of the 16th Intl. Conference on Natural Language Processing*, pages 36–44
Hyderabad, India, December 2019. 

statistical models, with less than 12,000 event mentions in each. Secondly, to the best of our knowledge, we present the first ever baseline for neural event detection in Hindi using this model. Our architecture uses both word and character embeddings and captures information from them distinctly, before combining them into a coherent representation of both. This is then used to determine the label for each input word. The proposed architecture is language invariant as well, such that no part of the system undergoes a change when training on different languages. In presenting this architecture, we highlight the importance of using subword level information in order to incorporate morphological as well as syntactic features in event extraction. This can also be extended to other semantically oriented sequence labeling tasks

## 2 Related Work

Neural approaches to sequence tagging are common due to extensive developments in named entity recognition. Huang et al. (2015) introduced and cultivated the use of bidirectional LSTMs to incorporate features that could be used for sequence tagging using a CRF. Ma and Hovy (2016)'s architecture and the NeuroNER program (Dernoncourt et al., 2017) provided a basic architecture and influenced multiple developments to most sequence labeling tasks, including event detection and extraction (Araki, 2018). The task of event extraction in any language involves the identification of the event nugget (Ahn, 2006). Prominent work has been done to analyze the lexical and semantic features of event representation (Li et al., 2013), which served as a basis for neural event nugget detection (Liang et al., 2017).

The task of neural event detection has been attempted using a combination of networks, but mostly revolving around the use of convolutional neural architectures. Work in this approach focused on various aspects such as max-pooling to retrieve the structure of event nugget information (Nguyen and Grishman, 2015), modeling the skip-gram architecture to learn lexical feature representations (Chen et al., 2015) as well as using dynamic CNNs in order to extract lexical and syntactic features in parallel (Nguyen and Grishman, 2016). Recurrent neural architectures have also been employed for this task, which predict the location of the trigger based on combining the forward and backward features of sentences in which events occur (Nguyen et al., 2016; Ghaeini et al., 2016). Note that in both cases architectures focused on dealing with structural, lexical and contextual features.

In the domain of multi-lingual and cross lingual event detection, Feng et al. (2018) uses a combination of both LSTMs and CNNs for creating a language independent architecture for capturing events, while Goud et al. (2019a) used stacked RNNs for sequence labeling and a language discriminator to learn language features. The latter architecture implements the use of the character embeddings, but does not identify the relevant features independent of the word embeddings.

## 3 Model Description

In this section, we describe the ALINED model for the event detection. Primarily, we focus on how to capture event representation at both a character and a word level. In this model, we had to focus on the following major considerations:

1. Syntactic and lexical information captured by previous event detection tasks should be accounted for.

2. Furthermore, sub-word information is essential as morphological features are also useful in identifying event semantics if the language is morphologically rich, or has a free word order structure.

Fundamentally, our architecture generates character embeddings through convolution and aggregates this information using bidirectional LSTMs (Hochreiter and Schmidhuber, 1997). The same is done over pretrained word embeddings in parallel, creating distinct intermediate representations. These representations are combined using a highway architecture for a final representation, which is used for the sequence tagging task.

### 3.1 Generating Contextual Character Embedding

In order to generate character embeddings from the input sentence, we first use a CharCNN (Kim et al., 2016). Let $\mathcal{C}$ be the dictionary of all the characters in the language and $\mathcal{V}$ be all the words in the language. We first define the character embeddings matrix $\boldsymbol{E} \in \mathbb{R}^{d \times |\mathcal{C}|}$, where $d$ is the dimensionality of the character embeddings, with the constraint that $d < |\mathcal{C}|$. Let word $w_i \in \mathcal{V}$

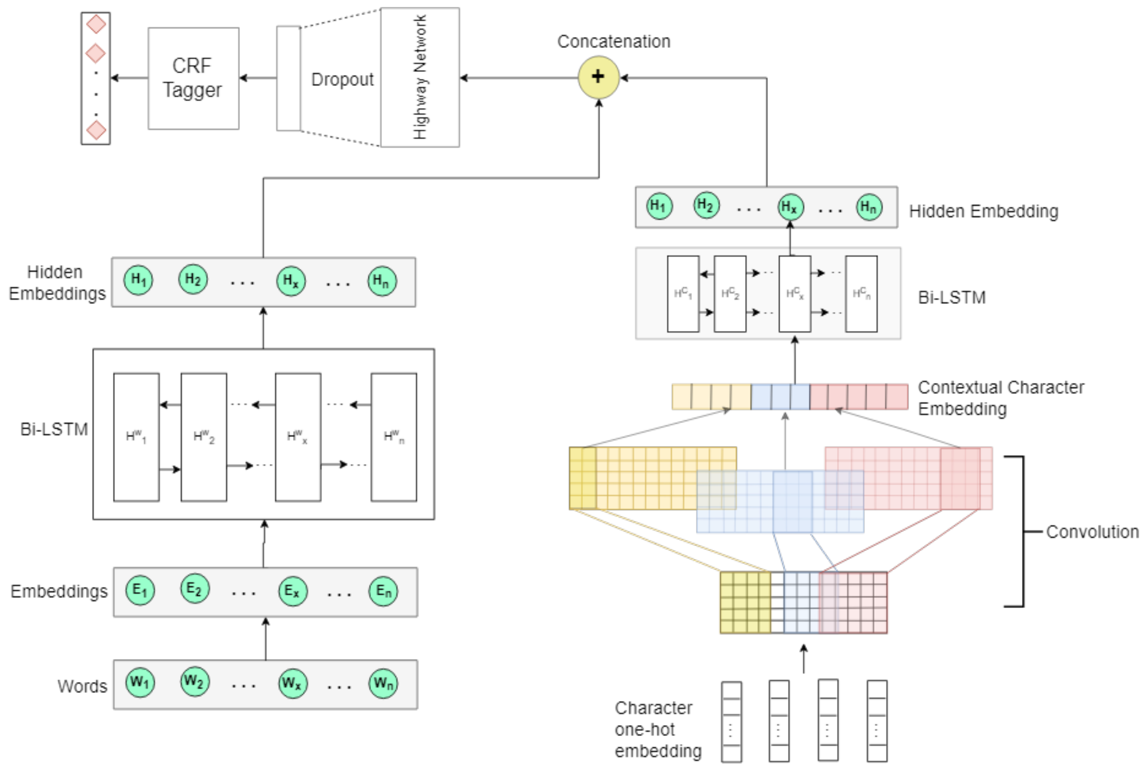


Figure 1: The proposed ALINED model

be made up of $n$ characters, such that $\boldsymbol{c}^{w_i} = [c_1^{w_i}, c_2^{w_i}, \ldots, c_n^{w_i}]$. The character representation of $w_i$ is therefore given by $\boldsymbol{E}^{w_i} \in \mathbb{R}^{d \times n}$.

We define a filter $\boldsymbol{W} \in \mathbb{R}^{d \times b}$ where $b$ is the width of the filter. We apply a narrow convolution between $\boldsymbol{E}^{w_i}$ and $\boldsymbol{W}$, to obtain the embedding of $w_i$ as:

$$e_i^{w_i} = f(\boldsymbol{W} \cdot \boldsymbol{E}^{w_i}[*, i : i + b - 1]) + b \quad (1)$$

where $\boldsymbol{E}^{w_i}[i : i + b - 1]$ accounts for all the characters of given window size of the word. The obtained embedding $e^{w_i} \in \mathbb{R}^{n-b+1}$. The function $f$ is a non-linear function such as a hyperbolic tangent or a sigmoid. It is applied over the Frobenius inner product of the filter and the embedding value as $\boldsymbol{A} \cdot \boldsymbol{B} = \mathrm{Tr}(\boldsymbol{A}\boldsymbol{B}^T)$ for any two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$.

We use max-pooling over the output embedding (instead of mean-pooling as it better incorporates the nature of natural language sequences (Xiang et al., 2016)) as:

$$w_i^c = \max_i e_i^{w_i} \quad (2)$$

For a total of $h$ filters, each of varying widths, we get different representations of $w_i$. Therefore $\boldsymbol{w}_i^c = [\boldsymbol{w}_1^c, \boldsymbol{w}_2^c, \ldots, \boldsymbol{w}_h^c]$ is the representation of the $i$th word.

The aggregated word representations based on character information now capture the features that represent the event semantics at a sub-word level accurately. However, the contextual information has not been accounted for yet. This is done by using a bidirectional LSTM, as mentioned above.

$$h_i^c = \text{bi-LSTM}(\boldsymbol{w}_i^c, h_{i-1}^c, h_{i+1}^c) \in \mathbb{R}^{k \times l} \quad (3)$$

The bi-LSTM hidden state vector $\boldsymbol{h}^c = [h_1^c, h_2^c, ..., h_k^c]$, each $h_i^c$ of dimension $\mathbb{R}^l$ is now propagated to the rest of the network. $\boldsymbol{h}^c$ can be seen as a lexically context-aware character representation of the words of the input sentence.

### 3.2 Using Contextual Word Embeddings

To capture structural information well, we use contextual word embeddings. Let $\boldsymbol{w} = [w_1, w_2, ..., w_k]$ be the words in a sentence. Let their corresponding pre-trained word embeddings be $\boldsymbol{e}^w = [e_1^w, e_2^w, ..., e_k^w]$. We aggregate the meaning of the sentence by passing the word embeddings through a bidirectional LSTM layer, as follows:

$$h_i^w = \text{bi-LSTM}(\boldsymbol{e}_i^w, h_{i-1}^w, h_{i+1}^w) \in \mathbb{R}^{k \times l} \quad (4)$$

Now each hidden state of $\boldsymbol{h}^w = [h_1^w, h_2^w, ..., h_k^w]$, i.e., each $h_i^w$ of dimension

$\mathbb{R}^l$, is used in the rest of the network. Since the pre-trained word embeddings are already contextual in nature, we do not process it further. Note that $\boldsymbol{h}^w$ can be seen as the semantically context-aware representation of the words of the input sentence. This also includes the structure of event representation in that sentence.

### 3.3 Combining Character and Word Representations

Given the representations of the hidden states from characters and words, we combine the two using a concatenation function followed by a highway network. The concatenation is represented as follows:

$$h_i = f(h_i^w, h_i^c) \tag{5}$$

The function $f(\cdot)$ is the concatenation function, which can be represented as:

$$f(h_i^w, h_i^c) = \begin{cases} h_i^w \odot h_i^c & (6) \\ \boldsymbol{W} \cdot h_i^w \odot (1 - \boldsymbol{W}) \cdot h_i^c & (7) \\ \boldsymbol{W}^w \cdot h_i^w \odot \boldsymbol{W}^c \cdot h_i^c & (8) \end{cases}$$

Equation 6 is a direct concatenation of the hidden states $\boldsymbol{h}^c$ and $\boldsymbol{h}^w$. A direct concatenation automatically implies that the information gathered from the representations are given equal weight. However, this is not true for all languages, as languages with fewer inflections require less information from the character representations and more from the word representations.

Equations 7 and 8 attempt to account for this by using a shared weight concatenation and a weighted concatenation respectively. In equation 7, $\boldsymbol{W} \in \mathbb{R}^{k \times k}$ is a weight matrix, where the values are scaled down to 1, in order to capture the relative importance of each $h_i^c$ and $h_i^w \, \forall h_i^c \in \boldsymbol{h}^c, h_i^w \in \boldsymbol{h}^w$. This shared weighting is a modification of the concept of *leaky integration* (Bengio et al., 2013). On the other hand, equation 8 uses two independent weight matrices, $\boldsymbol{W}^c, \boldsymbol{W}^w \in \mathbb{R}^{k \times k}$, which does not constrain the network to use on other the other hidden representation. However, the gradients are still clipped at a low value ($\approx 1$) to avoid explosion.

We then use the highway network (Srivastava et al., 2015) on the combined hidden state vector $\boldsymbol{h}$. This network adaptively "carries" some dimensions of $\boldsymbol{h}$ to the output for predicting the correct label sequence. Therefore, the hidden states undergo the following transformation (Wen et al., 2016):

$$h_i = \rho(h_i) \odot g(\boldsymbol{W}_H \cdot \bar{h}_i + b_H) + (1 - \rho(h)) \odot \bar{h}_i \tag{9}$$

The function $\rho(h^w) = \sigma(\boldsymbol{W}_\rho \cdot h_i + b_\rho)$, which is a simple activation function. $g$ is any non-linear function, such as sigmoid or hyperbolic tangent. Following the highway network's output, we pass the hidden embeddings to a dropout layer, which effectively reduces the number of hidden units by a fraction $d$, so $\boldsymbol{h}_{drop} \in \mathbb{R}^{k/d \times l}$, and a linear layer, which maps the $\boldsymbol{h}_{drop}$ to a smaller embedding space. We label this space $\boldsymbol{h} \in \mathbb{R}^{k/d \times f}$ ($f$ being the dimensions of the feature space) for brevity.

### 3.4 Sequence Tagging Layer

In the sequence tagging layer, we use the combined embeddings to identify the most likely sequence of tags for the input sentence. With the aggregated combined hidden state $\boldsymbol{h}$, we have the information required to assign tags to the words of the input sentence. For this, we use conditional random fields (CRF). The traditional formulation of a CRF can be written, given a set of observations sequences $X = x_1, x_2, ..., x_k$ and sequence of labels $Y = y_1, y_2, ..., y_k$ as,

$$p(Y|X; W, b) = \frac{\prod_{i=1}^{k} \exp(y_{i-1}, y_i, X)}{\sum_{y' \in \mathcal{L}} \prod_{k}^{i=1} \exp(y'_{i-1}, y'_i, X)} \tag{10}$$

where $\mathcal{L}$ is the set of possible labels in the tagset.

Since the observation sequence in our formulation is essentially the output vector $\boldsymbol{h}$, we can simplify the above equation by performing softmax to score the likelihood of a label being assigned. Therefore, the probability distribution is computed as,

$$P(y_i = t | h_i) = \frac{\exp(h_i^T w_j + b_j)}{\sum_k \exp(h_i^T w_m + b_m)} \tag{11}$$

with $j, m \in \mathcal{L}$ as tag labels. We also compute the transition probability $T$ of the label $y_i$ being assigned to $h_i$ given the labels of $h_{i-1}$. Therefore, the probability of the sequence of labels over the hidden states can be computed as:

$$Seq(Y, \boldsymbol{h}) = \sum_{i=1}^{k} P(y_i = t | h_i) + \sum_{i=1}^{k} T(y_i = t | y_{i-1} = t');\ t, t' \in \mathcal{L} \tag{12}$$

Therefore the probability of that sequence $Y$ computed above is calculated as:

$$p(Y|\boldsymbol{h}) = \frac{\exp\left(Seq(Y, \boldsymbol{h})\right)}{\sum_{y' \in \mathcal{L}} \exp\left(Seq(y', \boldsymbol{h})\right)} \qquad (13)$$

## 4 Experimental Setup

In this section, we go over the various experiments, implementation details such as number of epochs, training time, datasets and the like. These are covered in detail for the replicability of our results, which are highlighted in section 5.

### 4.1 Datasets

To train and evaluate our model, we use the following datasets for each of the languages we work with multiple corpora, as our experiments span multiple languages.

1. The TempEval-3 TimeBank dataset was used for English (UzZaman et al., 2012). The corpus consists of 61,418 tokens for training and 6,756 event mentions.

2. For Spanish, we use the ModeS TimeBank (Modern Spanish TimeBank 1.0) (Nieto and Saurí, 2012) for training and testing. This was used in SemEval-2013 Task 1 Task B (UzZaman et al., 2013). The corpus consists of 57,977 tokens.

3. For Italian, we use Ita-TimeBank's ILC corpus (Caselli et al., 2011a) the Italian corpus annotated using ISO-TimeML rules for events and temporal information. The corpus consists of 68,000 tokens and 10,591 event mentions.

4. For French, we use the French TimeBank as it is the ISO-TimeML annotated reference corpus for event annotation tasks (Bittar et al., 2011). The corpus consists of 16,208 tokens and 2,100 event mentions.

5. For Hindi, we use the gold-standard corpus of Goud et al. (2019b), which consists of 810 event annotated news articles based on modified TimeML rules. The dataset has 242,201 tokens and 20,190 event mentions.

### 4.2 Model Implementation and Training Details

The datasets are annotated in the IOB format. At a word level, B represents the first token of an event, I represents all the other tokens of an event and O represents the tokens which are not a part of any event in the sentence. We train the model for 50 epochs, but the loss tends to stabilize at 25 to 35 epochs. We use a 40 dimensional character embedding, which we create ourselves, as mentioned in section 3.1. The CNN uses 40 filters with a window size of 3.

For our contextual word embeddings, we use fastText embeddings for English (Bojanowski et al., 2017) which are pretrained on commonCrawl and the Wikipedia corpus. FastText embeddings are also used for Hindi, French, Spanish and Italian word representations (Grave et al., 2018). The bi-LSTM trains on a fixed 300 hidden dimensions for all the bi-LSTMs in the architecture.

For the linear and dropout layers, the dropout is fixed to 0.3. The initial learning rate parameter is 0.015, which increases with a momentum of 0.9. On approaching the end of an epoch, the learning rate decays at a rate of 0.05. We train on a negative log-likelihood loss function

## 5 Results and Analysis

In this section, we analyze the results of the ALINED model, and compare them to the current state of the art systems for the various languages we train on. We also provide a rigorous error analysis of our system and methodology.

Since no single system has compared work in event detection across the five languages that we have chosen for the experiments here, we draw comparisons to the various systems that trained on the individual or group of languages that have been used. Table 1 ahows the direct comparison of results.

1. For English, we compare our system to the SemEval-2013 Task 1 Task B (UzZaman et al., 2013), detection of event extents. We compare our models' scores with those of the best performing models of SemEval-2013.

2. SemEval-2013 Task 1 Task B (UzZaman et al., 2013) performs the task of detecting event extents in Spanish texts. We compare our model performance to FSS-TimeEX and TipSemB-F, the best performing models in that task.

3. Caselli et al. (2011b) establishes the current state of the art for data driven models in temporal and event extent information in Italian.

| Language | Model | Precision | Recall | F1-Score |
|---|---|---|---|---|
| English | ATT-1 (Jung and Stent, 2013) | 81.44 | 80.67 | 81.05 |
| | ATT-2 (Jung and Stent, 2013) | 81.02 | 80.81 | 80.91 |
| | ATT-3 (Jung and Stent, 2013) | **81.95** | 75.57 | 78.63 |
| | KUL (Kolomiyets and Moens, 2013) | 80.69 | 77.99 | 79.32 |
| | **ALINED** | 78.79 | **87.00** | **82.70** |
| Spanish | FSS-TimEX (Zavarella and Tanev, 2013) | 89.80 | 42.40 | 57.60 |
| | TIPSemB-F (UzZaman et al., 2013) | **91.70** | **86.00** | **88.80** |
| | **ALINED** | 86.77 | 83.22 | 84.96 |
| Italian | TIPSemIT_basic (Caselli et al., 2011b) | 90.00 | 77.00 | 83.00 |
| | TIPSemIT_FPC5 (Caselli et al., 2011b) | 89.00 | 81.00 | 85.00 |
| | TIPSemIT_FPC5Sem (Caselli et al., 2011b) | **91.00** | **83.00** | **87.00** |
| | **ALINED** | 79.92 | 81.85 | 80.87 |
| French | CRF-kNN (Arnulphy et al., 2015) | **87.00** | 79.00 | **83.00** |
| | Bittar (2009) | 46.00 | **82.00** | 64.00 |
| | **ALINED** | 84.48 | 67.12 | 74.81 |
| Hindi | **ALINED** | **78.22** | **76.08** | **77.13** |

Table 1: Comparison of Model Performance

The system is a modification of the TipSem system. We compares our models to their reported scores. However, the corpus used in Caselli et al. (2011b) is the Ita-TimeBank which has been augmented with further annotations and resources, while our system uses just the Ita-TimeBank for event extraction.

4. For French, we did not find systems that did event extraction from the French TimeBank corpus. The existing literature either creates and evaluates on a modified corpus (Bittar, 2009) or provides annotations trained on the TimeML annotated data and tested on Fr-TempEval2) (Arnulphy et al., 2015). Therefore, we compare our performance to those, while also understanding that the comparison is not a strict metric. We hope to establish the scores here as baseline for further improvement over models in event detection in French.

5. To the best of our knowledge, there is no baseline system available for event detection in Hindi, therefore, we provide our model as the first performance metric in that direction.

In most comparisons, our models perform equally well or better than the current systems for each of the above languages. we do not annotate or augment any of our data sources for using this model, so the reference corpora are being trained and tested upon, which are mentioned in section 4.1.

The calculation of the metrics of comparison, precision, recall and accuracy are calculated as follows:

$$precision = tp/(tp + fp)$$
$$recall = tp/(tp + fn)$$
$$f - measure = 2 * (P * R)/(P + R)$$

where $tp$ is a true positive, where the part of the extent identified in the system output is the same as the expected output, $fp$ is a false positive, where the token identified as part of the extent by the system is not a part of the expect output, and $fn$ is a false negative, where a token not identified as a part of the extent by the system output, is a part of the expected output.

We note a lower precision score in case of English and Spanish, as the number of false positives are slightly higher. We attribute this difference to the fact that due to the combination of sub-word level features, the model seems to sometimes ”spill over” the boundary of single word or nominal. However, higher recall implies that there are fewer false negatives, meaning the model more accurately identifies those words which are in the event span. More labeled data would be very useful in learning the span boundaries, especially for nominal events, as the network would have more samples to learn the variations in the methods of event representation.

For English, surprisingly, we see that an increase in the F1-scores. We attribute this to a combination of factors, including well defined verbal affixes which are attributed to events, and effective weighted combination of character and word embeddings.

For Italian, we train and test solely on the Ita-TimeBank, whereas the current state of the art system trained on an augmented Ita-TimeBank (Caselli et al., 2011b), which was enriched with more labeled data. Similarly, in French, we use the established French TimeBank, while experiments in French so far have been on self-annotated (Arnulphy et al., 2015) or TimeML corpora (Bittar, 2009). Since these repositories of augmented data were not available to us at the time of writing this paper, the values reflect the same. However, it is to be noted that our system does provide an accuracy that is close to the currently reported state-of-the-art even in the absence of language specific features, explaining the fact that sub-word information is necessary for event detection in Italian and French as well.

For Hindi, our architecture provides a good baseline. However, the training data consists of far too many words that are out of vocabulary, which is a major issue in working with word embeddings. While the concatenation of sub-word information mitigates this, a system focused on a better representation of out of vocabulary words would significantly help the network. However, this required a larger labeled corpus as well, which makes this a challenge as Hindi is a low-resource language in terms of corpora for event detection and extraction.

## 6 Conclusion

In this paper, we show the development of ALINED, a language invariant neural sequence tagging architecture for event detection in five different languages, namely, English, Spanish, Italian, French and Hindi. We develop insight into the use of sub-word level information and combining it effectively. with the lexical and syntactic information.

For our training and testing, we use only established corpora, which have not been augmented or changed in any way. We perform almost at par or better then the current state of the art in all the languages we train in. We establish a new best F-score for event extraction in English. We also establish the baseline for training and testing on the French TimeBank and for event extraction as a task in Hindi.

Our model has been thoroughly error-analyzed, which we have explained based on the comparison of system output and expected tags. Given the nature of our results, we aim to establish the importance of sub-word level information in event detection. Further work in this task could be done by providing augmented reference corpora, so that problems based on lack of labeled data do not limit further research in this topic. This could also be tackled by effectively introducing transfer learning to neural event detection, where the model learns the representation of events irrespective of language, while accounting for sub-word, lexical and structural information.

## References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Jun Araki. 2018. *Extraction of Event Structures from Text*. Ph.D. thesis, Ph. D. thesis, Carnegie Mellon University.

Béatrice Arnulphy, Vincent Claveau, Xavier Tannier, and Anne Vilnat. 2015. Supervised machine learning techniques to detect timeml events in french and english. In *International Conference on Applications of Natural Language to Information Systems*, pages 19–32. Springer.

Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. 2013. Advances in optimizing recurrent networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8624–8628. IEEE.

André Bittar. 2009. Annotation of events and temporal expressions in french texts. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 48–51. Association for Computational Linguistics.

André Bittar, Pascal Amsili, Pascal Denis, and Laurence Danlos. 2011. French timebank: an iso-timeml annotated reference corpus. In *Proceedings of the 49th Annual Meeting of the Association for*

Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 130–134. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Tommaso Caselli, Valentina Bartalesi Lenzi, Rachele Sprugnoli, Emanuele Pianta, and Irina Prodanof. 2011a. Annotating events, temporal expressions and relations in italian: the it-timeml experience for the ita-timebank. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 143–151. Association for Computational Linguistics.

Tommaso Caselli, Hector Llorens, Borja Navarro-Colorado, and Estela Saquete. 2011b. Data-driven approach using semantics for recognizing and classifying timeml events in italian. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 533–538.

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.

Hai Leong Chieu and Yoong Keok Lee. 2004. Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–432. ACM.

Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. 2017. Neuroner: an easy-to-use program for named-entity recognition based on neural networks. *EMNLP 2017*, page 97.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, page 1.

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

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.

Goran Glavaš and Jan Šnajder. 2014. Event graphs for information retrieval and multi-document summarization. *Expert systems with applications*, 41(15):6904–6916.

Jaipal Goud, Pranav Goel, Allen J. Antony, and Manish Shrivastava. 2019a. Leveraging multilingual resources for open-domain event detection. In *Proceedings 15th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pages 76–82.

Jaipal Goud, Pranav Goel, Alok Debnath, Suhan Prabhu, and Manish Shrivastava. 2019b. A semantico-syntactic approach to event-mention detection and extraction in hindi. In *Proceedings 15th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pages 63–75.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Hyuckchul Jung and Amanda Stent. 2013. Att1: Temporal annotation using big windows and rich syntactic and semantic features. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 20–24.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Oleksandr Kolomiyets and Marie-Francine Moens. 2013. Kul: data-driven approach to temporal parsing of newswire articles. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 83–87.

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.

Dongyun Liang, Weiran Xu, and Yinge Zhao. 2017. Combining word-level and character-level representations for relation classification of informal text. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 43–47.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1064–1074.

Teruko Mitamura, Zhengzhong Liu, and Eduard H Hovy. 2015. Overview of tac kbp 2015 event nugget track. In *TAC*.

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.

Marta Guerrero Nieto and Roser Saurí. 2012. Modes timebank 1.0. *Linguistic Data Consortium (LDC), Philadelphia, PA, USA*.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56.

James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. 2010. Iso-timeml: An international standard for semantic annotation. In *LREC*, volume 10, pages 394–397.

James Pustejovsky, Jessica Littman, Roser Saurí, and Marc Verhagen. 2006. Timebank 1.2 documentation. *Event London, no. April*, pages 6–11.

Roser Saurı. 2010. Annotating temporal relations in catalan and spanish timeml annotation guidelines. Technical report, Technical Report BM 2010-04, Barcelona Media.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Naushad UzZaman, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333*.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 1–9.

Y Wen, R Luo, J Wang, et al. 2016. Learning text representation using recurrent convolutional neural network with highway layers. In *Neu-IR: The SIGIR 2016 Workshop on Neural Information Retrieval*, volume 2016. Association for Computing Machinery (ACM).

Yang Xiang, Xiaoqiang Zhou, Qingcai Chen, Zhihui Zheng, Buzhou Tang, Xiaolong Wang, and Yang Qin. 2016. Incorporating label dependency for answer quality tagging in community question answering via cnn-lstm-crf. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1231–1241.

Vanni Zavarella and Hristo Tanev. 2013. Fss-timex for tempeval-3: Extracting temporal information from text. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 58–63.