ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

# Learning Rich Event Representations and Interactions for Temporal Relation Classification

Onkar Pandit[1], Pascal Denis[1] and Liva Ralaivola[2]

1- MAGNET, Inria Lille - Nord Europe, Villeneuve d'Ascq, France
onkar.pandit@inria.fr, pascal.denis@inria.fr

2- QARMA, IUF, LIS, Aix-Marseille University, CNRS, Marseille, France,
Criteo AI Labs, Paris, France. liva.ralaivola@lif.univ-mrs.fr

**Abstract**. Most existing systems for identifying temporal relations between events heavily rely on hand-crafted features derived from event words and explicit temporal markers. Besides, less attention has been given to automatically learning contextualized event representations or to finding complex interactions between events. This paper fills this gap in showing that a combination of rich event representations and interaction learning is essential to more accurate temporal relation classification. Specifically, we propose a method in which i) Recurrent Neural Networks (RNN) extract contextual information ii) character embeddings capture morpho-semantic features (e.g. tense, mood, aspect), and iii) a deep Convolutional Neural Network (CNN) finds out intricate interactions between events. We show that the proposed approach outperforms most existing systems on the commonly used dataset while using fully automatic feature extraction and simple local inference.

## 1 Introduction

Recovering temporal information from texts is a crucial part of language understanding, and it has applications such as question answering, summarization, etc.

Temporal relation identification is divided into two main tasks, as identified by TempEval campaigns [1]: i) the identification of EVENTs and other time expressions (the so-called TIMEX's), and ii) the classification of temporal relations (or TLINKs) among and across events and time expressions [2, 3, 4, 5]. Possible relations for this latter task include temporal precedence (i.e., BEFORE or AFTER), *inclusion* (i.e., INCLUDES or IS_INCLUDED), and others inspired from Allen's algebra. In this work, we concentrate on temporal relation classification, specifically EVENT-EVENT relations.

The problem becomes harder in the absence of explicit temporal connectives (e.g., *before*, *during*), determining temporal relations depends on numerous factors, ranging from tense and aspect to lexical semantics and even world knowledge. To address this issue, most state-of-the-art systems for EVENT-EVENT classification [2, 3, 5] rely on manually-crafted feature sets directly extracted from human annotations, complemented with syntactic features, and semantic features extracted from static knowledge bases like WordNet or VerbOcean. Such an approach is tedious, error-prone and the semantics of events is poorly modelled due to lack of coverage of existing lexical resources and blindness to event contexts.

We here propose a radically different approach where we altogether dispense with hand-designed features and instead learn task-specific event representations. These representations include information both from the event words *and* its surrounding context, thus giving access to the events' arguments and modifiers. Furthermore, character based

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

model capture another type of information captured by morphology such as tense and aspect of event. Plus, we also attempt to learn the potentially rich interactions between events. Concretely, our learning framework, as depicted in Fig.1, is based on a neural network architecture, wherein: i) Recurrent Neural Network (RNN) is employed to learn contextualized event representations ii) character embeddings is used to capture morphological information, hence encoding tense, mood and aspect information, and iii) a deep Convolutional Neural Network (CNN) architecture is then used to acquire complex, non-linear interactions between these representations.

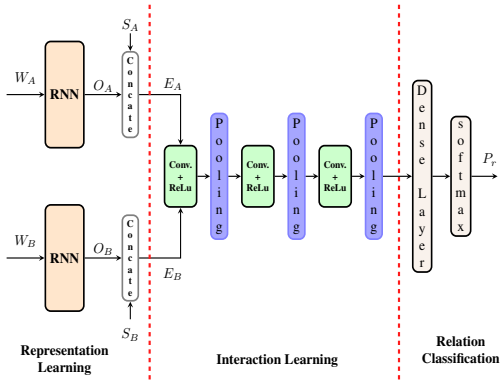


Fig. 1: Architecture of our proposed model.

This is one important step up from the recent work of Mirza and Tonelli [4], which simply uses pre-trained word embeddings for event words and still have to resort to additional hand-engineered features to achieve good temporal classification accuracy. We show our system based on fully automatic feature extraction and interaction learning outperforms other local classifier systems.

## 2 Related Work

Recent temporal classification systems use machine learning techniques due to the availability of annotated datasets. Earlier work [2, 6] studied *local* models (i.e., making pairwise decisions on pairs of events) and used gold-standard features extracted from TimeML annotations. State-of-the-art local models such as ClearTK [7] relied on an enlarged set of predicted features, and classifiers. These *local* models may generate globally incoherent temporal relations, in the sense that the symmetry and transitivity holding between relations are not followed at the document level. This has led to development of *global* models [3]. The state-of-the-art method [5] proposes a structured prediction approach.

These methods all rely on manually engineered features, which fail to model the semantics of events. To address this issue, [4] have evaluated the effectiveness of pre-trained word embeddings of event head-word. They also demonstrated the potency of basic vector combination schemes. However, representing events with word embeddings of only its head-word is not effective and important contextual information is lost. Recently proposed LSTM-based neural network architectures [8, 9] learn event representation with use of event head word as well as context. Also, newly proposed method [10] have shown efficacy of context with use of gated RNN-attention based neural architecture. However, by using only word embeddings they fail to capture inflectional morphology of event head word, which includes crucial linguistic information such as tense, aspect, and mood. Also they lacked in finding complicated interaction between events and relied only on concatenation of event features. Moreover they [9] used syntactically parsed trees as inputs to the LSTM which adds burden of pre-processing.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

## 3 Method

Our proposed neural architecture (Fig.1), consists of two main components: Representation Learning and Interaction Learning. In the Representation Learning part, a bag-of-words in a fixed size window centred on each event word is fetched and fed into a RNN to get more expressive and compact representation of events. Output of the RNN is concatenated with character embedding of event head-word to get the final vector for each event. This vector representation is then used at the Interaction Learning stage: the vector representation of each event is fed to a convolution layer and the final pooling layer outputs an interaction vector between the events. A dense layer is used to combine the interaction vector before obtaining a probability score for each relation.

### 3.1 Representation Learning

**Context-aware Representation** Each word is encoded in the event-word window with word embeddings [11]. As a result, each word is assigned a fixed $d$-dimensional vector representation. Let $c_l$ be the context length for each event head word $w_t$. Thus we consider a window of $2c_l + 1$ words as input to the RNN. We represent this as matrix $W = [\mathbf{w_{t-c_l}} \cdots \mathbf{w_t} \cdots \mathbf{w_{t+c_l}}] \in \mathcal{R}^{(2c_l+1)\times d}$. Note that while considering event context we stop at sentence boundary, also special symbols are padded if context is less than $c_l$. The relation between input and output of RNN at each time $t$ is given as follows,

$$h_t = \sigma_h(Q_h w_t + U_h h_{t-1} + b_h) \tag{1}$$

$$o_t = \sigma_o(Q_o h_t + b_o) \tag{2}$$

where, $w_t$ is the word embedding vector provided at each time step, $h_t$ is hidden layer vector and $o_t$ is output vector. $Q$, $U$, and $b$ are weight matrices and vector; $\sigma_h$ and $\sigma_o$ are activation ReLU functions. For a given event, the $o_{t+c_l}$ output vector captures a complete information about the whole sequence. The outputs of the RNN networks give compact representations $O_A$ and $O_B$ of the events.

**Morphological Representation** Semantics and arguments of events are captured with context-aware representation but it doesn't capture morphological information such as tense and aspect. For instance, tense information is captured from context as well with auxiliary verbs (*will*, *is*). However in the absence of these words, tense information of event is expressed in inflectional suffixes such as *-ed*, *-ing*. To obtain this information, event head word $w_t$ is represented as a sequence of character *n-grams* $c_1, \cdots, c_m$ where $m$ is number of *n-grams* in word. Fixed $d_c$ dimension vector is learned for each *n-gram*. Eventually morphological representation of word is obtained by adding vectors of *n-grams* present in the word. Standard *fasttext* [13] method is used to get this representation.

Context-aware vector $O_A$ and character embedding vector $S_A$ are concatenated to obtain final event representation as $E_A$, similarly $E_B$ also obtained for event $B$.

### 3.2 Interaction Learning

A deep Convolution Neural Network (CNN) is employed to learn nonlinear interactions from $E_A$ and $E_B$. It is comprised of three convolution and pooling layers placed alternatively. We feed concatenated learned event representations

$$E_{AB} = E_A \oplus E_B \tag{3}$$

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

to the first convolution layer, where $\oplus$ is the concatenation operation. Each convolution layer $i$ use filters $f_i$, after what we compute a feature map

$$m_i^k = \sigma(f_i.E_{AB} + b_i) \qquad \forall k \in \{1, 2, 3\}, \tag{4}$$

where $f_i$, $b_i$ are filters and bias matrices respectively and $\sigma$ is the ReLU activation. The output is down-sampled with a max-pooling layer to keep prominent features intact. The output of the last layer gives the interaction between $A$ and $B$ ($\rho$ is max-pooling).

$$E_{comb} = \rho(m_i^3) \tag{5}$$

The combined $E_{comb}$ vector is fed to a fully connected dense layer, followed by a softmax function to get a probability score for each temporal relation class.

# 4 Experiments

## 4.1 Datasets and Evaluation

**Relations** Following recent work [5], reduced set of temporal relations: *after, before, includes, is_included, equal, vague* are considered for classification.

**Evaluation** Complying with common practice, system's performance is measured over gold event pairs (pairs for which relation is known). Our main evaluation measure is the Temporal Awareness metric [12], adopted in recent TempEval campaigns. We also used standard precision, recall, and F1-score.

**Datasets** We used TimeBank (TB) and AQUAINT (AQ) dataset for training, TimeBank-Dense(TD) for development and Platinum (TE3-PT) dataset for testing. These are the most popular datasets used for the task which have been provided at TempEval3 [1].

## 4.2 Training Details

We used pre-trained Word2Vec vectors from Google[1]. Each word in the context window of event is represented with this 300-dimension vector. Character embeddings for event head word are obtained from *fasttext* [13] which is also 300-dimension vector. Note that only one RNN is trained with weight sharing to learn representation. Hyperparameters were tuned on the development set using a simple grid search. Considered values are: window size ($c_l$: 3,4,5), number of neurons at RNN (#RNN: 64,128,256,512), number of filters for CNN (#filters: 32,64,128,256), dropout at input (0.1,0.2,0.3,0.4). We also explored a number of optimization algorithms such as AdaDelta, Adam, RMSProp and Stochastic Gradient Descent(SGD). Optimal hyper-parameter values are $c_l = 4$,#RNN =256, #filters = 64, dropout = 0.4 and Adam optimizer.[2] Once we got the optimal parameter values from the validation set, we re-trained multiple models with 50 different seed values on the combined training and development data and report the averaged test performances.

## 4.3 Results

We first compare our RNN-Deep CNN approach to various baseline systems to assess the effectiveness of the learned event representations. We also want to disentangle the respective role of the representation learning and interaction learning.

**Baseline systems** As Mirza and Tonelli [4] reported results only with pairwise *f-score* on different dataset, we re-implemented their system with scikit-learn Logistic

[1] `https://code.google.com/archive/p/word2vec/`

[2] We tried different unidirectional and bidirectional variations of RNN-LSTM and GRU, but RNN gave the best development results.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

| | Systems | Pair Classification | | | Temporal Awareness | | |
|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 |
| (a) | $W_A \oplus W_B$ | 39.3 | 34.2 | 35.5 | 27.1 | 45.8 | 34.1 |
| | $O_A \oplus O_B$ | 35.7 | 38.9 | 37.2 | 36.5 | 35.9 | 36.2 |
| | $E_A \oplus E_B$ | 37.6 | 44.5 | 40.8 | 41.2 | 45.9 | 43.4 |
| (b) | $MLP(E_A, E_B)$ | 40.2 | 46.3 | 43.0 | 51.8 | 42.5 | 46.7 |
| | $CNN(E_A, E_B)$ | 38.2 | 53.7 | 44.6 | 41.7 | 60.1 | 49.2 |
| | $DCNN(E_A, E_B)$ | 39.4 | 58.9 | 47.2 | 43.2 | 70.1 | 53.4 |
| (c) | ClearTK | - | - | - | 33.1 | 35.0 | 34.1 |
| | LSTM | 38.7 | 43.1 | 40.5 | 34.6 | 51.7 | 41.4 |
| | SP | - | - | - | 69.1 | 65.5 | 67.2 |

Table 1: Results of baseline and state-of-the-art systems

regression module, using $l_2$ regularization (noted $W_A \oplus W_B$). Word embeddings $W_A$ and $W_B$ of events $A$ and $B$ obtained from Word2Vec were simply concatenated (as this is their best performing system). We conducted number of experiments to obtain optimal parameters and reported best performing system's results. We got lower than the originally reported results as the dataset is sparsely annotated compared to the one in the paper (TimebankDense). As additional baselines, we used our Representation Model to learn $O_A$ and $O_B$, subsequently $E_A$, $E_B$, but combined these vectors with simple concatenation ($O_A \oplus O_B$ and $E_A \oplus E_B$). In another setting learned representation is combined with multi-layer perceptron (MLP) and single-layer convolution (CNN).

**Comparison to Baseline Systems** Sections $a$ and $b$ of Table 1 summarize the performance of these different systems in terms of pairwise classification accuracy and temporal awareness scores. Looking at the first two rows of the table, we see that, as hypothesized, contextually rich features outperform pre-trained event head word embeddings when combined with simple concatenation, both in pairwise classification and in temporal awareness. A further gain in the performance with character embeddings shows the effectiveness of morphological features. Results from the section establish the effectiveness of our rich representation learning. In section $b$ of the table, we present results of the system with different interaction learning.The system with MLP interaction learning outperforms simple *concatenation* ($E_A \oplus E_B$) system, demonstrating importance of interaction learning. Leveraging both rich event representations learning and non-linear interaction learning yield the best scores overall, cf. $CNN(E_A, E_B)$ and $DCNN(E_A, E_B)$, which shows their complementarity. There, the Deep CNN outperforms the single-layer CNN, with F1 scores of $53.4$ and $49.2$, respectively. This confirms the importance of non-linear interaction learning in the task.

**Comparison with State-of-the-art** Finally, in the section $c$, we compare the performance of our best system, $DCNN(E_A, E_B)$, with recently proposed systems : $W_A \oplus W_B$ [4], ClearTK [7], which was the winner of the TempEval 2013 campaign, the structured prediction (SP) approach[5], which is the best system to date and recently proposed LSTM [14]. Our system delivers substantial improvements over $W_A \oplus W_B$, ClearTK, and LSTM based system. However our system lags in comparison with SP as it relies only on simple local inference opposed to global inference at learning step, also totally dispenses with hand-crafted features. It is important to observe that our system closes the performance gap between SP system and automatic feature learned systems.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

## 5 Conclusion and Future Work

In this work, we proposed RNN based neural architecture to learn event representation and CNN model to get interaction between events. A new perspective towards the combination of events proved to be effective in getting compound interactions. We compared result of our system with multiple baselines and state-of-the art systems and shown effectiveness. We now plan to learn features and interaction while considering global consistency in the relations of event pairs.

## Acknowledgement

This work was supported by ANR Grant GRASP No. ANR-16-CE33-0011-01, as well as by a grant from CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020.

## References

[1] N. UzZaman, H. Llorens, L. Derczynski, J. Allen, M. Verhagen, and J. Pustejovsky. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. ACL, 2013.

[2] I. Mani, M. Verhagen, B. Wellner, C. M. Lee, and J. Pustejovsky. Machine learning of temporal relations. ACL, 2006.

[3] N. Chambers, T. Cassidy, B. McDowell, and S. Bethard. Dense event ordering with a multi-pass architecture. *TACL*, 2014.

[4] P. Mirza and S. Tonelli. On the contribution of word embeddings to temporal relation classification. COLING, 2016.

[5] Q. Ning, Z. Feng, and D. Roth. A structured learning approach to temporal relation extraction. EMNLP, 2017.

[6] Nathanael Chambers, Shan Wang, and Dan Jurafsky. Classifying temporal relations between events. ACL, 2007.

[7] S. Bethard. ClearTK-TimeML: A minimalist approach to tempeval 2013. ACL, 2013.

[8] D. Dligach, T. Miller, C. Lin, S. Bethard, and G. Savova. Neural temporal relation extraction. ACL, 2017.

[9] F. Cheng and Y. Miyao. Classifying temporal relations by bidirectional lstm over dependency paths. ACL, 2017.

[10] Y. Meng and A. Rumshisky. Context-aware neural model for temporal information extraction. ACL, 2018.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. NIPS, 2013.

[12] N. UzZaman and J. F. Allen. Temporal evaluation. ACL, 2011.

[13] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin. Advances in pre-training distributed word representations. In *LREC*, 2018.

[14] Y. Meng, A. Rumshisky, and A. Romanov. Temporal information extraction for question answering using syntactic dependencies in an lstm-based architecture. *CoRR*, abs/1703.05851, 2017.