Event Time Extraction with a Decision Tree of Neural Classifiers

Nils Reimers†, Nazanin Dehghani‡*, Iryna Gurevych†
† Ubiquitous Knowledge Processing Lab (UKP) and Research Training Group AIPHES
Department of Computer Science, Technische Universität Darmstadt
‡ School of Electrical and Computer Engineering, University of Tehran
www.ukp.tu-darmstadt.de

Abstract

Extracting the information from text when an event happened is challenging. Documents do not only report on current events, but also on past events as well as on future events. Often, the relevant time information for an event is scattered across the document.

In this paper we present a novel method to automatically anchor events in time. To our knowledge it is the first approach that takes temporal information from the complete document into account. We created a decision tree that applies neural network based classifiers at its nodes. We use this tree to incrementally infer, in a stepwise manner, at which time frame an event happened. We evaluate the approach on the TimeBank-EventTime Corpus (Reimers et al., 2016) achieving an accuracy of 42.0% compared to an inter-annotator agreement (IAA) of 56.7%. For events that span over a single day we observe an accuracy improvement of 33.1 points compared to the state-of-the-art CAEVO system (Chambers et al., 2014). Without retraining, we apply this model to the SemEval-2015 Task 4 on automatic timeline generation and achieve an improvement of 4.01 points F_1-score compared to the state-of-the-art. Our code is publically available.[1]

1 Introduction

Knowing when an event happened is useful for a lot of use cases. Examples are in the fields of time-aware information retrieval, text summarization, automated timeline generation, and automatic knowledge base population. Many facts in a knowledge base are only true for a certain time period, for example the presidency of a person. Hence, the population of a knowledge base can highly benefit from high quality event and event time[2] extraction (Surdeanu, 2013).

Inherent to events is the connection to time. Allan (2002) defines an event as "*something that happens at some specific time and place*". The challenges for automatic event time extraction are manifold. The temporal information in news articles which states when an event happened is, in most cases, not in the same or in neighboring sentences with the event (Reimers et al., 2016). It can be mentioned far before the event or far after the event. Even worse, for more than 60% of events, the specific day at which the event happened is not mentioned. However, from the world knowledge and causal relations, the reader can infer a lot of temporal information about those events and can often infer that the event happened before or after some specific point in time.

In this paper we describe a new classifier for automatic event time extraction. We use the TimeBank-EventTime Corpus (Reimers et al., 2016) to train and evaluate our proposed architecture. In contrast to other corpora on temporal relations, the annotation of the TimeBank-EventTime Corpus does not make restrictions where, and in which form, temporal information for an event must be provided. The annotators were allowed to take the whole document into account and were asked to answer, to the best of their ability, the question at which date or time period the event happened.

The event time annotation for some sample events is shown in the following:

- He was [**sent**]$_{1980\text{-}05\text{-}26}$ into space on May 26,

*During author's internship in the research training group AIPHES at UKP Lab, TU Darmstadt.

[1] https://github.com/ukplab/tacl2017-event-time-extraction

[2] We will refer to the temporal information when an event happened as *event time*.

Transactions of the Association for Computational Linguistics, vol. 6, pp. 77–89, 2018. Action Editor: Patrick Pantel.
Submission batch: 6/2017; Revision batch: 10/2017; Published 2/2018.

1980. He [**spent**]$^{\text{endPoint =1980-06-01}}_{\text{beginPoint=1980-05-26}}$ six days aboard the Salyut 6 spacecraft.

- [...] two areas [expected]$^{\text{endPoint =before 1998-02-06}}_{\text{beginPoint=before 1998-02-06}}$ to be hardest [**hit**]$_{\text{after 1998-01-01 before 1998-01-31}}$ when the effects of the Asian crisis [...].

This annotation imposes several challenges for an automatic approach:

1. The number of possible labels is infinite, as date values are part of the labels.
2. Due to the diverse types of events and due to varying temporal information for events, the structure of the labels varies.
3. Temporal information from the whole document must be taken into account.
4. For 12.6% of the events, the event time label is a combination of several temporal clues. An example could be that the annotator combined that the person went missing on the 15th and that the person went missing in the month of August. However, nowhere in text is the 15th of August explicitly mentioned.

The main contribution of this paper is the proposal of a novel combination of a decision tree combined with neural network classifiers for the nodes to solve the afore-mentioned challenges. To our knowledge, this is the first system that works on the complete document and can extract long-range relations between events and temporal expressions. Further, it is the first system that focuses on extracting begin and end points for events that span over multiple days.

Evaluated on the TimeBank-EventTime Corpus (Reimers et al., 2016), it achieves an accuracy of 42.0% compared to an inter-annotator agreement (IAA) of 56.7%. Compared to the state-of-the-art CAEVO system (Chambers et al., 2014), we observe a substantial improvement in accuracy of 33.7 percentage points for events that happened on a single day. For Multi-Day Events, we observe an accuracy of 24.3% using a strict metric.

We show that the proposed model generalizes well to new tasks and textual domains. We applied it without re-training to the SemEval-2015 Task 4 on automatic timeline generation. There, it achieves an improvement of 4.01 points F_1-score compared to the state-of-the-art.

2 Related Work

We start with a review on common annotation schemes to capture temporal information for events in documents. Afterwards, we present related work on automatically extracting temporal information for events.

2.1 Annotation of Events and Temporal Information

One of the most widely used specifications for events and temporal expressions is TimeML (Saurí et al., 2004). It provides specifications for the annotation of *events*, *temporal expressions*, and the *temporal links (TLINK)*. An *event* is defined as term for situations that happen or occur. *Temporal expressions*, such as times, dates, or durations, are annotated and their temporal values are normalized using the definitions of Ferro (2002). A TLINK is the relation between two events, between an event and a temporal expression, or between two temporal expressions. TimeML defines 14 different relation types, however, most corpora which are using the TimeML specification restrict the number of relations to a smaller set.

A prominent corpus using the TimeML specifications is the TimeBank Corpus (Pustejovsky et al., 2003), which was also the basis for the three shared tasks TempEval-1 (Verhagen et al., 2007), TempEval-2 (Verhagen et al., 2010) and TempEval-3 (UzZaman et al., 2013).

A drawback of TLINKs is the quadratic growth of possible TLINKs with the number of events and temporal expressions, resulting in more than 10,000 possible TLINKs for several documents in the TimeBank Corpus. As the annotation of such a large number of TLINKs would be impractical, annotation of those is always restricted in some form. For the TimeBank Corpus, only salient TLINKs were annotated. Which links are salient isn't well defined and a low agreement between annotators can be observed. The three TempEval shared tasks tried to improve the coverage and added some further temporal links for mentions in the same sentence. More dense annotations were applied by Bramsen et al. (2006), Kolomiyets et al. (2012), Do et al. (2012) and Cassidy et al. (2014). While Bramsen et al., Kolomiyets et al., and Do et al. only annotated some temporal links, Cassidy et al. annotated all Event-Event, Event-Time, and Time-Time

pairs in the same sentence as well as in the directly succeeding sentence leading to the densest annotation for the TimeBank Corpus. They used six different relation types: BEFORE, AFTER, INCLUDES, IS_INCLUDED, SIMULTANEOUS, and VAGUE, where VAGUE encodes that the annotators were not able to make a statement on the temporal relation of the pair.

2.2 Existent Event Time Extraction Systems

Most automatic approaches use the previously introduced TLINKs to train and evaluate systems for extracting temporal information about events. For a new document, the system first extracts the temporal relations between events and temporal expressions. In a post-processing step, those TLINKs are used to retrieve the information when an event happened.

Extracting the relations is often formulated as a pair-wise classification task. Each pair of events and/or temporal expressions is examined and classified according to the available relation classes. Ensuring transitivity is a big challenge when formulating this task as a pair-wise classification task. One simple but nonetheless frequently used solution is to automatically infer all temporal relations that can be derived from transitivity. Some systems have tried to take advantage of global information to ensure transitivity using Markov logical networks or integer linear programming (Bramsen et al., 2006; Chambers and Jurafsky, 2008; Yoshikawa et al., 2009; UzZaman and Allen, 2010). However, the gains were minor.

Chambers et al. (2014) proposes the CAEVO-system, a sieve-based-architecture that blends multiple classifiers into a precision-ranked cascade of sieves. The system was trained and evaluated on the TimeBank-Dense Corpus and created a dense TLINK annotation for all pairs of events and/or temporal expressions in the same and in adjacent sentences. The code is publically available.[3]

A bottleneck of current systems is the limitation to TLINKs for pairs that are in the same or in adjacent sentences. According to Reimers et al. (2016) 28.3% of the events happen at the document creation time (DCT). For the remaining 71.7% of events, the event time must be inferred via TLINKs. However, for 58.7% of those events the most informative time expression[4] is not in the same nor in the previous/next sentence. In conclusion, for 42.1% of all the events in a text it would be necessary to take long-range TLINKs into account to correctly retrieve the event time. Extending existing systems to take long-range relations into account is difficult due to a lack of training and evaluation data.

3 Event Time Annotation

We use the TimeBank-EventTime Corpus (Reimers et al., 2016) to evaluate our architecture for automatic event time extraction. The TimeBank-EventTime Corpus does not use the concept of TLINKs, instead, for every event, the annotators were asked to anchor the event in time as precisely as possible.

The annotation distinguishes between events that happened on a *Single Day* and *Multi-Day Events* that span over multiple days. For *Single Day Events*, the annotators provide the day the event happened in the format *YYYY-MM-DD*. In the case the exact date is not mentioned in the document, the annotators were asked to anchor the event in time as precisely as possible using the annotation *before YYYY-MM-DD* and *after YYYY-MM-DD*. *Before* notes that the event must have happened before the stated date and *after* that the event must have happened after the date. A combination of *before* and *after* is possible.

For Multi-Day Events, the annotators were asked to provide the begin and the end point of the event. As for Single Day Events, they were allowed to use the *before* and *after* notation in the case the explicit begin/end point is not mentioned in the document.

The annotated corpus contains news articles and TV broadcast transcripts from various sources written mainly between January and April 1998. The shortest document has five sentences, while the longest has 63 sentences. A label distribution can be found in (Reimers et al., 2016).

4 Automatic Event Time Extraction

In this section we first present our hierarchical tree approach to automatically infer the event times in

[3] http://www.usna.edu/Users/cs/nchamber/caevo/

[4] The *most informative temporal expression* is defined as the temporal expression giving the reader the information at which date, or in which time frame, the event happened.

a document. In Section 4.3 we present two baselines that we use for comparison: the first uses dense TLINKs extracted by the CAEVO system and the second baseline is a reduced version of the presented tree approach.

4.1 Event Time Extraction using Trees

We use the tree structure depicted in Figure 1 to extract the event time for a given target event. The structure was inspired by how annotators labeled the data. When annotating the text, the first decision is typically whether the event is a *Single Day Event* or a *Multi-Day Event*. In the case that it is a Single Day Event, the next question is whether the event happened at the *Document Creation Time (DCT)* or not. As the annotated data comes from the news domain, a large set of events (48.28% of the Single Day Events) happened at the document creation time. In the case the event did not happen at DCT, then the annotator scanned the text to decide whether the date when the event happened is explicitly mentioned or not. If it is not mentioned, the annotator used the *before* and *after* notation to define the time frame when the event happened as precisely as possible. For Multi-Day Events, the process is similar to determine the begin and end point of the event.

The first classifier is a binary classifier to decide whether the event is a Single or a Multi-Day Event. In the case it is a Single Day Event, the next classifier decides the relation between the event and the Document Creation Time (DCT). In the case the event happened at DCT, the architecture stops. If the event happened before or after DCT, the next classifier is invoked, detecting which temporal expressions are relevant. For all relevant temporal expressions, it is then determined whether the event happened simultaneously, before, or after the temporal expressions. The final step (2.4) outputs a single event time by narrowing down the information it receives from the relation to DCT (2.1) and the pool of relevant temporal expressions and relations (2.3).

For Multi-Day Events the process is similar, however, the system must return the begin and the end points. The system runs three processes in parallel: it extracts the relations to relevant time expressions for the begin point (3.1.1 and 3.1.2); it extracts the relation to DCT (3.2) and; it extracts the relations to relevant time expressions for the end point (3.3.1 and 3.3.2). There are three possible relations between a Multi-Day Event and the DCT: the event started and ended before the DCT; it started and ended after the DCT; or it started before DCT and ended after DCT. This information is taken into account in step 3.1.3 and 3.3.3 when producing single begin point and end point information for the given event.

4.2 Local Classifiers

This section describes the different local classifiers applied in our tree structure. For all except the *Narrow Down* classifier, we used the Convolutional Neural Networks Architecture (Lecun, 1989) depicted in Figure 2. The *Narrow Down* classifier is a simple, hand-crafted, rule-based classifier described in Section 4.2.6.

4.2.1 Neural Network Architecture

We use the same neural network architecture with slightly different configurations for the different local classifiers. The architecture is depicted in Figure 2 and is described in the following sections.

The neural network architecture is based on the design proposed by Zeng et al. (2014), which can achieve state-of-the-art performance on relation classification tasks (Zeng et al., 2014; dos Santos et al., 2015). The neural network applies a convolution over the word representations and position embeddings of the input text followed by a max-over-time pooling layer. We call the output of this layer *Input Text Features*. Those Input Text Features are merged with the word embedding for the event and time expression token. The merged input is fed into a hidden layer using either the hyperbolic tangent $tanh(\cdot)$ or a rectified linear unit (ReLU) as activation function. The choice of the activation function is a hyperparameter and was optimized on a development set. The final layer is either a single sigmoid neuron, in the case of binary classification, or a softmax layer. To avoid overfitting, we used two dropout layers (Srivastava et al., 2014), the first before the dense hidden layer and the second after the dense hidden layer. The percentages of the dropouts were set as hyperparameters.

Word Embeddings. We used the pre-trained word embeddings presented by Levy and Goldberg (2014). The embedding layer of our neural networks maps each token from the input text to their respective word embedding. Out-of-vocabulary tokens are

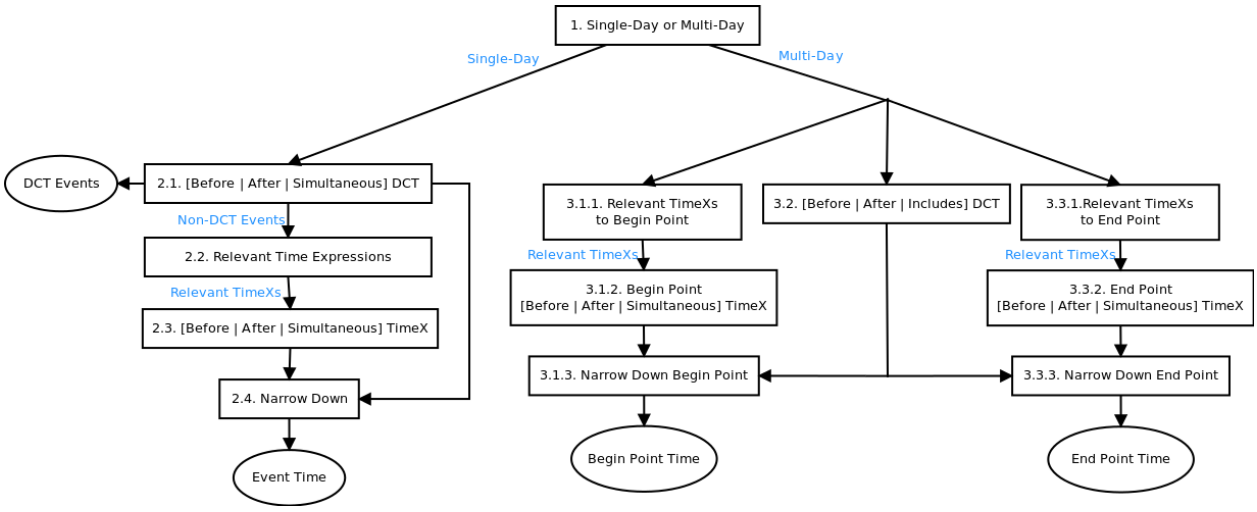

Figure 1: Tree structure used to extract the temporal information for an event. Rectangles are local classifiers based on deep convolutional neural networks except for the *Narrow Down* rectangles, which are simple rule based classifiers.

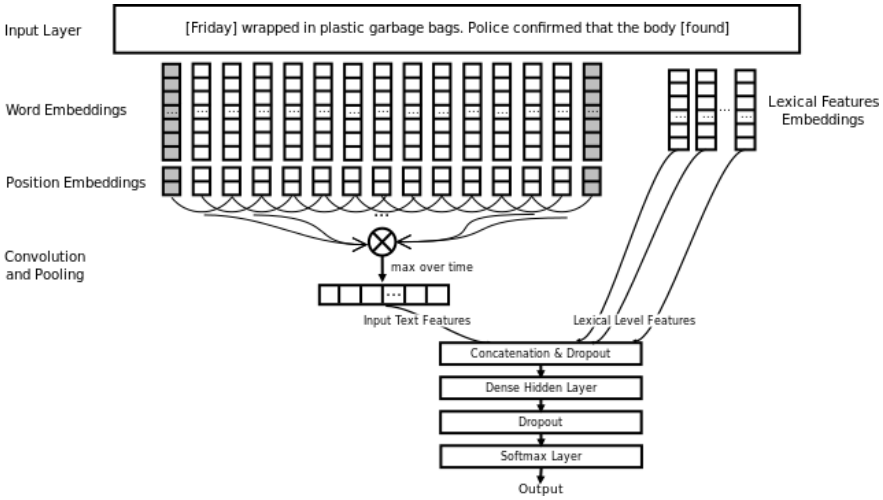

Figure 2: The neural network architecture used for the different local classifiers.

replaced with a special UNKNOWN token, for which the word embedding was randomly initialized.

Position Embeddings. Collobert et al. (2011) proposes the use of position embeddings to keep track how close words in the input text are to certain *target words*. For each input text, we specify certain words as targets. For example, we specify the event and the temporal expression as target words and train the network to learn the temporal relation between those. Each word in the input text is then augmented with the relative distances. Let pos_1, pos_2, ... be the positions of the target words in the input text. Then, a word at position j is augmented with the features $j - pos_1$, $j - pos_2$, $\cdots$. These augmented position features are then mapped in the embedding layer to a randomly initialized vector. The dimension of this vector is a hyperparameter of the network.

The word embeddings and the position embeddings are concatenated to form the input for the convolutional layer. In the case of two target words, the input for the convolutional layer would be:

$$emb_{output} = \{[we^{w_1}, pe^{1-pos_1}, pe^{1-pos_2}], [we^{w_2}, pe^{2-pos_1}, p^{2-pos_2}], ..., [we^{w_n}, pe^{n-pos_1}, pe^{n-pos_2}]\}$$

with we^{w_j} the embedding of the j-th word in the

input text, pe^{j-pos_k} the embedding for the distance between the j-th word and the target word k.

Convolutional & Max-Over-Time Layer. A challenge for the classifier is the variable length of the input text and that important information can be anywhere in the input text. To tackle this issue, we use a convolutional layer to compute a distributed vector representation of the input text. Let us define a vector x_k as the concatenation of the word and position embeddings for the position k as well as for m positions to the left and to the right:

$$x_k = ([we^{w_{k-m}}, pe^{k-m-pos_1}, pe^{k-m-pos_2}]||...||$$
$$[we^{w_k}, pe^{k-pos_1}, pe^{k-pos_2}]||...||$$
$$[we^{w_{k+m}}, pe^{k+m-pos_1}, pe^{k+m-pos_2}])$$

The convolutional layer multiplies all x_k by a weight matrix W_1 and applies the activation function component-wise. After that, a *max-over-time* is applied, i.e., the max-function is applied component-wise. The j-th entry of the convolutional and max-over-time layer output is defined as:

$$[conv_{output}]_j = \max_{1 \leq k \leq n} [\tanh(W_1 x_k)]_j$$

Lexical Features. Previous approaches heavily rely on lexical features. For example, the CAEVO system (Chambers et al., 2014) uses, for the classification of event-time edges, the token, the lemma, the POS tag, the tense[5], the grammatical aspect[6] and the class of event[7] as well as the parse tree between event and time expression. In our evaluation, we did not observe that these features have a significant impact on the performance. Hence, we decided to use the event and time tokens as the only features besides the dense vector representation of the input text. For multi-token expressions, we only use the first token.

Our architecture focuses on extracting the event time when event annotations and temporal expressions are provided. In order to evaluate the accuracy of this isolated step, we decided to use the provided annotations in the corpus. The baselines we compared against use these gold annotations as well.

Output. The distributed vector representation of the input text and the embeddings of event/time token are concatenated and passed through a dense layer. As the activation function, we allowed either the hyperbolic tangent or the rectified linear unit (ReLU). The choice is a parameter of the network. The final layer is either a single sigmoid neuron, in the case of binary classification, or a softmax layer to compute the probabilities of the different tags.

4.2.2 Single vs. Multi-Day Event Classification

The first local classifier, that decides whether an event is a *Single Day Event* or a *Multi-Day Event*, uses the event word as the target word.

4.2.3 DCT Classification

A Single Day Event can happen either before the document was created (`Before`-class), on the same day (`Simultaneous`-class), or it will happen at least one day after the document was created (`After`-class). The configuration of this local classifier is as in the previous section.

Note, to classify the relation to the DCT, in most cases, it was not important to know the concrete Document Creation Time. Therefore, we did not pass the DCT as a value to the network.

For Multi-Day Events, we decided to group the events into three categories: first, events that began and ended before the Document Creation Time (`Before`-class); second, events that began before DCT and ended after DCT (`Includes`-class); and third, events that will begin and end after DCT (`After`-class).

4.2.4 Detecting Relevant Time Expressions

In the case the event did not happen at the DCT, it is important to take the surrounding text and potentially the whole document into account to figure out at which date the event happened. For our classifier, we assume that temporal expressions are already detected in the document. To detect temporal expressions, tools like HeidelTime[8] can be used that achieve an F_1-score of 0.919 on extracting temporal expressions in the TimeBank Corpus (Strötgen and Gertz, 2015).

[5]Defined tenses: simple, perfect, and progressive

[6]Defined aspects in TimeBank: past, present, future

[7]Defined classes in TimeBank: occurrence, perception, reporting, aspectual, state, i_state, i_action

[8]`https://github.com/HeidelTime`

As an intermediate step to detect when an event happened, we first decide whether the temporal expression is relevant for the event or not. We define a temporal expression to be relevant, if the (normalized) value of the temporal expression is part of the event time annotation. The value of the temporal expression can either be the event time, or it can appear in the `before` or `after` notation.

The classifier is executed for all event and temporal expression pairs. The input text for the distributed text representation is the text between the event and the temporal expression.

4.2.5 Temporal Relation Classification

Given the relevant temporal expression for an event from the previous step, the next local classifier establishes the temporal relation between the event and the temporal expression. For a given, relevant event-temporal expression pair, it outputs `BEFORE` - when the event happened before the temporal expression, `AFTER` - when it happened after, or `SIMULTANEOUS` - when it happened on the mentioned date. This local classifier has the same configuration as the network used to detect relevant temporal expression.

4.2.6 Narrow Down Classifier

The goal of the Narrow Down Classifier, that is used in step 2.4, 3.1.3 and 3.3.3 in Figure 1, is to derive the final label given the information on the relevant temporal expressions, their relation to the event, and the relation to the document creation time. For most events in the corpus, this information was unambiguous, e.g., only one temporal expression was classified as relevant for the event. The proposed approach returns multiple relevant temporal expressions only for a small fraction of events. However, this number was too small to train and to validate a learning algorithm for this stage. Hence, we decided to implement a straightforward, rule-based classifier. This classifier is depicted in Algorithm 1.

It takes all relations to relevant temporal expressions as well as the relation to the Document Creation Time to derive the final output. In the case a `SIMULTANEOUS` relation exists, the classifier stops and the appropriate temporal expression is used as event time. If no such relation exists, a frequency distribution of the linked dates and relations is created for `BEFORE` as well as for `AFTER` relations. For example, when the system extracts three relevant `BEFORE` relations of different mentions of `date1` throughout the text and two relevant `BEFORE` relations of different mentions of `date2`, then the system would choose `date1` as a slot-filler for the before property. If there are as many relevant `BEFORE` relations for `date1` as for `date2`, the system will choose the lowest date for the before property (line 13-18). For `AFTER` relations, we use the same logic, except that we choose the largest date (line 23).

Algorithm 1 Narrow Down Classifier

```
1:  function NarrowDown(times)
2:      fd_before, fd_after = FreqDistribution()
3:      for [relation, time] in times do
4:          if relation is SIMULTANEOUS then
5:              return time
6:          else if relation is BEFORE then
7:              fd_before.new_sample(time)
8:          else if relation is AFTER then
9:              fd_after.new_sample(time)
10:         end if
11:     end for
12:     //fd_before elements have the fields .num=#samples
    and .time=time value
13:     if fd_before.size > 0 then
14:         // find the largest number of samples of a time
15:         max_samples = fd_before.max(_.num)
16:         //take minimum over all times having max samples
17:         before_time = fd_before.filter(_.num ==
    max_samples).min(_.time)
18:     end if
19:     if fd_after.size > 0 then
20:         // find the largest number of samples of a time
21:         max_samples = fd_after.max(_.num)
22:         //take maximum over all times having max samples
23:         after_time = fd_after.filter(_.num ==
    max_samples).max(_.time)
24:     end if
25:     return after + after_time + before + before_time
26: end function
```

4.3 Baseline

We use two baselines to compare our system. As the first baseline, we use the system presented in Reimers et al. (2016). The baseline is based on the multi-pass architecture CAEVO introduced by Chambers et al. (2014) and extracts all TLINKs between event mentions and temporal expressions. The system by Chambers et al. applies multiple rules and trained classifiers to extract those TLINKs. The dif-

ferent stages are ranked by precision and are executed consecutively. A shortcoming of the system is that it does not produce temporal information if an event lasted for more than a day. Hence, the system cannot be used to distinguish between Single Day and Multi-Day Events, nor can it extract the begin/end points for Multi-Day Events.

Our previously presented baseline uses the extracted relations for Single Day Events and generates a set of <relation, time> tuples in which the event is involved. We use the narrow down classifier from section 4.2.6 to extract the final label. When all extracted relations are of type `VAGUE`, the baseline returns that it cannot infer the time for the event.

The second baseline is a reduced version of the hierarchical tree. For this baseline, we first apply the classifier to decide whether it is a Single Day or Multi-Day Event. When it is a Single Day Event, we classify the relation to the document creation time (DCT) (classifier 2.1). When the event did not happen at DCT, we link it to the closest temporal expression in the document. For Multi-Day Events, we only run the classifier 3.2 to extract the relation to DCT. When the event happened before DCT, we set the begin and end point to `BEFORE DCT`; when it happened after DCT, we set both to `AFTER DCT`; and, when the relation was `Includes`, we set the begin point to `BEFORE DCT` and the end point to `AFTER DCT`.

5 Experimental Setup

We conduct our experiments on the TimeBank-EventTime Corpus (Reimers et al., 2016). The corpus is comprised of 36 documents and 1498 annotated events. We use the same split into training, development, and test set as Chambers et al. (2014) resulting in 22 documents for training, 5 documents for hyperparameter optimization, and 9 documents for the final evaluation. Using this split allows a fair comparison to the CAEVO system.

Hyperparameters for the individual local classifiers were chosen using random search (Bergstra and Bengio, 2012) with at least 1000 iterations per local classifier.

6 Experimental Results

We evaluate our system using two different metrics. The **strict metric** requires an exact match between the predicted label and the gold label. A disadvantage of this metric is that it does not allow partial agreement. The strict agreement between two annotators is fairly low for events where the exact date of the event was not mentioned.

In order to allow partial matches, we will also use a **relaxed metric**, which will judge two different labels only as an error, if those are mutually exclusive. Two labels are mutually exclusive, if there is no event date which could satisfy both labels at the same time. If the event happened on August 5th, 1998, the two annotations *before 1998-08-31* and *after 1998-08-01 before 1998-08-31* would both be satisfied. Therefore, these two different labels would be considered as correct. In contrast, the two annotations *after 1998-02-01* and *before 1997-12-31* can never be satisfied at the same time and are therefore mutually exclusive.

The score of the relaxed metric must be seen in combination with the strict metric. A system could trick the relaxed metric by returning a before date that is far in the future which results in a high relaxed score but a negligible strict score. Future research is necessary to judge the quality of different kind of partial matches and to design an appropriate metric.

6.1 System Performance

Following the recommendations in (Reimers and Gurevych, 2017), we train the system with 25 different random seed values, and compute the mean performance score and the standard deviation. Table 1 shows the results in comparison to the observed inter-annotator agreement (IAA). The inter-annotator agreement is based on two full annotations of the corpus. The chance-corrected agreement is $\alpha = 0.617$ using Krippendorff's α (Krippendorff, 2004). The two annotations were merged into a final gold label annotation of the corpus, which we used for training and evaluation.

The accuracy to distinguish between Single Day and Multi-Day Events is 78.2% on the test set, in comparison to an inter-annotator agreement of 81.8%. The overall performance is 42.0%, compared to an IAA of 56.7% using the strict metric.

For Multi-Day Events, we observe an accuracy with the strict metric of 24.5%, compared to an IAA of 52.0%. Breaking it down to the begin- and endpoint extraction, we observe a much lower accuracy for the begin point extraction of just 28.5%, com-

	System	IAA
Single vs. Multi-Day	78.2% ± 1.33	81.8%
Single Day (Strict)	74.6% ± 1.04	80.5%
Single Day (Relaxed)	92.5% ± 0.60	98.0%
Multi-Day (Strict)	24.5% ± 1.61	52.0%
Begin (Strict)	28.5% ± 0.73	63.8%
End (Strict)	66.5% ± 1.02	74.9%
Multi-Day (Relaxed)	74.6% ± 0.55	94.6%
Begin (Relaxed)	94.9% ± 0.38	98.6%
End (Relaxed)	80.2% ± 0.73	96.1%
Overall Acc. (Strict)	42.0% ± 1.21	56.7%
Overall Acc. (Relaxed)	84.6% ± 0.71	95.3%

Table 1: Accuracy for the different stages of our system in comparison to the observed inter-annotator agreement (IAA). The *strict* metric requires an exact match between the labels. The *relaxed* metric requires that the two annotations are not mutually exclusive.

pared to 66.7% accuracy for the end point extraction. However, using the relaxed metric, we see an accuracy of 94.9% for the begin point and 80.2% for the end point. We can conclude that the extraction of the begin point works well, however, in a large set of cases (66.7%) the extracted begin point is less precise than the gold annotation.

The baseline based on the CAEVO system from Chambers et al. (2014) can only be applied to Single Day Events, as TLINK types that define the start or the end of an event do not exist. We ran this baseline on all events that were correctly identified as Single Day Events. The performance of this baseline is depicted in Table 2. For the proposed approach we observe a performance increase from 41.2% to 74.6%. For 18.3% of the events, the retrieved label of the proposed approach was less precise than the gold label. An example of a less precise label would be *before 1998-12-31* while the gold label was *before 1998-08-15*. A clear wrong label was observed for 7.1% of the generated labels.

A big disadvantage of a dense TLINK annotation is the restriction of TLINKs for events and temporal expression that are in the same, or in adjacent, sentences. For 32.0% of the events, the baseline was not able to infer any event time information. As our system outputs a label for every event, we see a slightly increased number of wrong labels in comparison to the baseline.

Single Day Events	Ours	CAEVO
Exact match	74.6%	41.2%
Less precise	18.3%	21.5%
Wrong label	7.1%	5.4%
Cannot infer time	-	32.0%

Table 2: Distribution of the retrieved labels for the proposed system and for the baseline. *Less precise* are labels where the time frame when the event has happened is larger than for the gold label. *Wrong label* are labels which are in clear contradiction to the gold standard.

Table 3 compares the proposed system against the reduced tree that only classifies the type of the event (Single Day or Multi-Day) and the relation to the document creation time. We observe a significant drop in accuracy for Single Day Events, indicating that just classifying the relation to the document creation time is insufficient for this task.

System	SD	MD	Overall
Full system	74.6%	24.3%	42.0%
Reduced tree	40.4%	19.6%	24.2%
CAEVO	41.2%	-	18.1%

Table 3: Comparison of the accuracy (strict metric) for Single Day Events (SD), Multi-Day Events (MD) and overall. Reduced tree uses only the local classifiers 1, 2.1 and 3.2.

6.2 Error Analysis

Error propagation is an important factor in a decision tree. Table 4 depicts the accuracy of the different local classifiers. We compare those to a Majority Vote baseline. For all local classifiers we can see a large performance increase over the baseline. We observe the lowest accuracy for the classifiers of the begin point (3.1.1. and 3.1.2.). This is in line with the previous observation of the low accuracy for begin point labels as well as with the low IAA for begin point annotations.

The root classifier, which decides whether the event is a Single Day or a Multi-Day Event, is the most critical classifier. This classifier is responsible for 21.7% of the erroneously labeled events. However, with an accuracy of 78.3% it is already fairly close to the IAA of 81.6% and it is unclear if this classifier could substantially be improved.

	System	Majority Vote
1. Event Type	78.3%	54.5%
Single Day Event		
2.1. DCT Rel.	84.2%	55.6%
2.2. Relevant	79.1%	66.0%
2.3. Relation	81.0%	72.7%
Multi-Day Event		
3.1. Begin Point		
3.1.1. Relevant	79.0%	68.9%
3.1.2. Relation	63.1%	42.9%
3.2. DCT Rel.	65.2%	46.8%
3.3. End Point		
3.3.1. Relevant	83.8%	65.1%
3.3.2. Relation	85.1%	79.0%

Table 4: Accuracy for the different local classifiers vs. a Majority Vote baseline. Local classifiers are numbered as depicted in Figure 1.

As mentioned in the introduction, the annotators were not restricted to the dates that are explicitly mentioned in the document but could also create new dates. For example, in the sentence *It's the [second day]$_{date:1998\text{-}03\text{-}06}$ of an [offensive]$_{beginPoint=1998\text{-}03\text{-}05...}$* it is clear for the annotator that the offensive started on 1998-03-05. However, this date is not explicitly mentioned in the text, only the date 1998-03-06 is mentioned. We call such dates *out-of-document* dates. Handling those cases is extremely difficult and our system is currently not capable of creating such out-of-document dates. Table 5 depicts the error rate introduced by those dates.

As the table depicts, 12.6% of the event time labels are affected by out-of-document dates. An especially high percentage of such dates is observed for the begin point of Multi-Day Events. In a lot of these cases the document states either an explicit or a rough estimation on the duration of the event. In the previous example, the text stated that the offensive already lasted for two days. In another example, the document gives the information that the event started *in recent years* or that it lasted *for roughly 2 1/2 years*.

6.3 Ablation Test

Table 6 presents the changes in accuracy in percentage points when individual components of the proposed system are changed. We observe a slight drop of -2.3 percentage points if bidirectional LSTM-networks with 100 recurrent units are used instead of Convolutional Neural Networks. LSTM-networks showed for other NLP tasks state-of-the-art performance, however, for this task they were not able to improve the performance. One reason could be the comparably small training set of 22 documents. A further disadvantage of the BiLSTM-networks was the significantly longer training time, prohibiting running an extensive hyperparameter tuning.

	Out-of-document dates
Single Day Events	3.0%
Multi-Day Events	24.1%
Begin Point	17.0%
End Point	9.9%
Overall	12.6%

Table 5: Percentage of labels in the test set affected by out-of-document dates.

Configuration	Accuracy
Full system	42.0%
BiLSTM instead of CNN	-2.3
Rnd. word embeddings	-7.7
No input text feature	-9.7
No position feature	-3.9
No narrow down	-1.3

Table 6: Change in accuracy (strict metric) in percentage points when replacing individual components of the architecture.

An important factor for the performance was the pre-trained word embeddings. Replacing those with randomly initialized embeddings decreased the performance by -7.7 percentage points. As before, we think this is due to the small training size. A large number of test tokens do not appear in the training set and several tokens only appear infrequently in the training set. Hence, the network is not able to learn meaningful representations for those words.

Our system successfully uses the text between the event and the temporal expression (*Input Text Features*) for classifying the relation between those. Removing this part of the architecture decreases the accuracy by -9.7 percentage points. Further, it appears that not only the token itself, but also the position of the token relative to the event / time token is taken

into account. Removing this position information from the input text feature reduces the performance by -3.9 percentage points.

Replacing the narrow down classifier with a classifier that randomly selects one of the relevant temporal expressions reduces the performance by only -1.3 percentage points. For most events, there was only one relevant temporal expression extracted. We analyzed the parameter settings for the top five performing local classifiers for each stage. The activation function (*tanh* and *ReLU*) appears to have a negligible impact on the performance.

6.4 Event Timeline Construction

We evaluated our system on the shared task *SemEval-2015 Task 4: TimeLine: Cross-Document Event Ordering* (Minard et al., 2015). The goal is to construct an event timeline for a target entity given a set of 30 documents from Wikinews on certain topics. We use the setting of Track B, where the events are provided. We used HeidelTime to detect and normalize time expressions. We then ran our system out of the box, i.e., without retraining for the new dataset.

For the shared task, an event can occur either at a specific day, in a specific month, or in a specific year. Events that can not be anchored in time are removed from the evaluation. We implemented simple rules that transform our system output to the format of the shared task: if an event is simultaneous with a specific time expression, we will output this date. If our system returns that it happened before and after a certain date, it will output the year and month if both dates are in the same month. If both dates are in the same year but in different months, it will output the year. Events with predicted timespans of over more than one year are rejected. For Multi-Day Events, we only use the begin point as only this information was annotated for this shared task.

Two teams participated in the shared task (GPLSIUA and HeidelToul). Currently, the best published performance was achieved by Cornegruta and Vlachos (2016) with an F_1-score of 28.58. Our system was able to improve the total F_1-score by 4.01 points as depicted in Table 7.

A challenge for our system is the different anchoring of events in time: while our system can anchor events at two arbitrary dates, the SemEval-2015 Task 4 only anchors events either at a specific day, month or year. When our system returns the event time value *after 2010-10-01* and *before 2010-11-30*, we had to decide how to anchor this event for the generated timeline. For such an event, three final labels would be plausible: 2010-10-xx, 2010-11-xx, and 2010-xx-xx. A similar challenge occurs for events that received a label like *before 2010-11-30*. If we anchor it in 2010-11-xx, we must be certain that the event happened in November. Similarly, if we anchor it in 2010-xx-xx, we must be certain that the event happened in 2010. Such information cannot be inferred directly from the returned label of our system. As only 30 documents on a single topic were provided for training, we could not tune the transformation accordingly. A manual analysis revealed that this transformation caused around 15% of the errors.

System	Airbus	GM	Stock	Total
Our approach	**30.37**	**28.83**	**38.01**	**32.59**
Cornegruta	25.65	26.64	32.35	28.58
GPLSIUA_1	22.35	19.28	33.59	25.36
HeidelToul_2	16.50	10.94	25.89	18.34

Table 7: Performance of our system on the SemEval-2015 Task 4 Track B for the topics Airbus, General Motors, and stock market.

7 Conclusion

Event Time Extraction is a challenging classification task as the set of labels is infinite and the label depends on the information that is scattered across the document. The presented classifier is able to take the whole document into account and to infer the date when an event has happened. We applied the system to the TimeBank-EventTime Corpus and achieved an accuracy of 42.0% in comparison to an inter-annotator agreement of 56.7% using a strict metric. For 74.6% of the Single Day events, the exact event time could be extracted. This is a 33.1 percentage points improvement in comparison to the state-of-the-art approach by Chambers et al. (2014).

We demonstrated the generalizability by applying it to the SemEval-2015 Task 4 on timeline generation, where it improved the F_1-score by 4.01 percentage points compared to the state-of-the-art.

Acknowledgements

This work has been supported by the German Research Foundation as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) under grant No. GRK 1994/1. We would like to thank the TACL editors and reviewers for their effort and the valuable feedback we received from them.

References

James Allan. 2002. Topic Detection and Tracking: Event-based Information Organization. pages 1–16. Kluwer Academic Publishers, Norwell, MA, USA.

James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.*, 13:281–305, February.

Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. 2006. Inducing Temporal Graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 189–198, Stroudsburg, PA, USA. Association for Computational Linguistics.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An Annotation Framework for Dense Event Ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 501–506, Baltimore, Maryland, USA. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 698–706, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense Event Ordering with a Multi-Pass Architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Savelie Cornegruta and Andreas Vlachos. 2016. Timeline extraction using distant supervision and joint inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1936–1942.

Quang Xuan Do, Wei Lu, and Dan Roth. 2012. Joint Inference for Event Timeline Construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 677–687, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cícero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying Relations by Ranking 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 of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 626–634.

Lisa Ferro. 2002. TIDES. Instruction Manual for the Annotation of Temporal Expressions. Technical report, MITRE TECHNICAL REPORT.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2012. Extracting Narrative Timelines As Temporal Dependency Structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 88–97, Stroudsburg, PA, USA. Association for Computational Linguistics.

Klaus Krippendorff. 2004. *Content Analysis: An Introduction to Its Methodology (second edition)*. Sage Publications.

Yann Lecun, 1989. *Generalization and network design strategies*. Elsevier.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 302–308.

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Ruben Urizar. 2015. SemEval-2015 Task 4: TimeLine: Cross-Document Event Ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 778–786.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. 2003. The TIMEBANK Corpus. In *Proceedings of Corpus Linguistics 2003*, pages 647–656, Lancaster, UK.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in*

Natural Language Processing (EMNLP), pages 338–348, Copenhagen, Denmark, September.

Nils Reimers, Nazanin Dehghani, and Iryna Gurevych. 2016. Temporal Anchoring of Events for the TimeBank Corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, volume 1: Long Papers, pages 2195–2204. Association for Computational Linguistics, August.

Roser Saurí, Jessica Littman, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. 2004. TimeML Annotation Guidelines, Version 1.2.1.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January.

Jannik Strötgen and Michael Gertz. 2015. A Baseline Temporal Tagger for all Languages. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 541–547, Lisbon, Portugal, September. Association for Computational Linguistics.

Mihai Surdeanu. 2013. Overview of the TAC 2013 Knowledge Base Population Evaluation: English Slot Filling and Temporal Slot Filling. In *Proceedings of the TAC-KBP 2013 Workshop*, Gaithersburg, Maryland, USA.

Naushad UzZaman and James F. Allen. 2010. TRIPS and TRIOS System for TempEval-2: Extracting Temporal Information from Text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 276–283, Stroudsburg, PA, USA. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, Leon Derczynski, Marc Verhagen, James F. Allen, and James Pustejovsky. 2013. SemEval-2013 Task 1: TempEval-3: Evaluating Time Expressions, Events, and Temporal Relations. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Gerogia, USA.

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. SemEval-2007 Task 15: TempEval Temporal Relation Identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 75–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marc Verhagen, Roser Saurí, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 Task 13: TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 57–62, Stroudsburg, PA, USA. Association for Computational Linguistics.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly Identifying Temporal Relations with Markov Logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, ACL '09, pages 405–413, Stroudsburg, PA, USA. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation Classification via Convolutional Deep Neural Network. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014*, pages 2335–2344, Dublin, Ireland.