Evolution Discovery in Textual Data

Event Ordering is a very important field in Event Extraction. Multiple approaches are available that aim to provide a better understanding of the temporal relationships between events within a text but the relationships extracted are complex in interpretation and representation point of view or the models lack in identifying enough relationships. In this paper, we present a timeframe-based event ordering to enable a simple event ordering model that enables a temporal relationship extraction limited between events happening in the same period of time, regardless of the duration of that period. Those events are grouped in timeframes and the timeframes are connected through temporal relationships. This approach aims to enable an evolution discovery for textual event elements.

Keywords: *Event Ordering, Event Extraction, Temporality Evaluation, Textual Knowledge Discovery*

Introduction

Knowledge has had multiple definitions over the centuries. We used two main definitions that led us to our context-awareness approach. On one hand, Aristotle defined knowledge as the result of a reasoning process based on the observation made on the exterior world (Johansen, 2021). On the other hand, Bachimont's definition (Bachimont et al., 2002) provided in the knowledge engineering field presents knowledge as the meaning given to a specific symbol in a particular context (based on a specific referent). These definitions show the necessity of a context-awareness approach in assigning meaning and identifying the activity. Knowledge discovery (Frawley et al., 1992) aims at identifying previously unknown knowledge from several sources and representing it using ontologies and knowledge bases. To extract knowledge from textual data (Hu, 2020), most approaches work at the sentence level where sentences are dealt with individually without using the whole text while analyzing a specific sentence. Another approach requires a huge amount of annotated data to enable the use of models such as BERT (Devlin et al., 2019). Most of these techniques do not consider the context of knowledge production, such as activity sector and domain environments.

This approach is a part of a context-awareness approach (Matta et al., 2021) that aims to enhance knowledge discovery in texts. We defined context elements such as actors, objects, locations, environments, domain, and the temporality dimension and presented possible ways to extract them. In the context-awareness perspective, the evolution of the context can be distributed in multiple categories: static or dynamic, discrete or continuous. Those categories reflect the temporality dimension. Having a need to analyze temporality in the text we focused on the Event Extraction field and the Event Ordering where huge work on temporal relation for events mentioned in a text was provided. In our work, we aim to enable textual knowledge extraction using a context-awareness approach that enables a global view of the text while using context elements as a reference to enrich the knowledge extraction. In this paper, we will use the timeframe approach (Matta et al., 2022) to improve event extraction and event ordering in text in order to enable an evolution discovery of an event within a text or multiple texts. We will start by presenting the related work, then the timeframe types and extraction. Later, provide our approach to enable an evolution discovery in textual data. Finally, the result and evaluation will be followed by the conclusion.

Related Work

This section will be divided into two main segments: Event Extraction and Event Ordering. We will go through multiple different approaches available before selecting the one we will be focusing on. More importantly, we will define which elements in the text will be considered an event.

Event Extraction

An event is an occurrence of a happening (Moens & Steedman, 1988; Hogenboom et al., 2016). It may be related to a specific date or time, place, people, objects, or other events. Event Extraction is one of the most important branches of Knowledge Extraction from textual data. Event-oriented models provide effective requirements to build decision support systems or question answering models (Duan et al., 2018). This branch started in the health care field for bacterial behavior analysis (Y. Chen et al., 2020) and with the progressive advancement of big data, the event extraction field gained popularity to enable information extraction from the exponentially growing heterogeneous textual data collections. From social media posts to web pages, news messages, articles, or researchers, event-oriented knowledge discovery enables event identifications along with the related entities that play a role in those events (Mejri & Akaichi, 2017; Zhou et al., 2020).

There are two types of event extraction (Xiang & Wang, 2019): Open-domain and Closed-domain event extraction. Open-domain event extraction is a non-specific event detection. It enables the detection of events and their arguments based on their syntactic and semantic role. This method is used for story segmentation, first story detection, topic detection… It is also considered a data-driven approach (Hogenboom et al., 2016) where researchers, using statistical methods and data mining, aim to convert data into knowledge. While closed-domain event extraction is the extraction of specific, previously defined events of interest. Those events are relatively important to the domain of application. For example, event detection in police reports can be accidents or homicide events. The place, time, victims, perpetrators, etc., are specifically named arguments that are relevant to specific events. Another example can be in the financial field where economic event analysis (Jacobs & Hoste, 2022) can be used for business intelligence decision support models. This approach is used for event mention detection, event trigger, event argument, and causal relation event analysis. Multiple approaches can be used for closed-domain event extraction. There is the pipeline approach (Liu et al., 2021) that distributes the extraction onto multiple classification stages. It goes as follows: first, identification of the event trigger based on keyword matching. Second, an argument classifier based on syntactic and semantic dependencies to identify the phrase segment that is classified as an argument. Lastly, assigning a role to the identified arguments. Other than the pipeline approach, we can find a join-based approach where the events type and its arguments are classified simultaneously in order to avoid errors generated by the trigger classification set. The reinforcement learning approach and the incremental learning model (Li et al., 2022) can be applied to identify the role of different arguments related to a specific event in the same sentence. Deep-learning methods (Li et al., 2021) are based on annotated datasets and transformers but their performances are relative to the quantity of annotated data taken as input.

Knowledge Graph construction based on event extraction is one of the potential representations for the events and their entity extraction (Liu et al., 2020). Two representation types can be considered, entity-centric knowledge graphs and event-centric knowledge graphs. Entity-centric considers all entities as nodes while event-centric has only events as nodes and the entities connected to the events are added inside the nodes for better visualization of all the events. In our approach, we will only be considering temporal relations and ignore all other arguments. This will imply that the representation will be similar to the event-centric representation.

Event Ordering

The temporal analysis is very important as it enables the study of causal, consequence, or impact effects of events and happenings (Ning, Feng, & Roth, 2019). Comprehending temporal relations is a key factor for storyline construction, time-slot filling, or narrative processing. It is also important for the question answering models since there are answers that change with time while other answers have a permanent nature (Duan et al., 2018).

Temporal event relation extraction has two main approaches: a data-driven approach and a hybrid approach that uses constraint-based and data-driven approaches (Lim et al., 2019). Data-driven models are trained using annotated datasets, for event ordering the join-based approach is the most commonly used. A constraint-based approach (Leeuwenberg & Moens, 2017) enables applying inferences in the processing and generation of information. A basic example of inferences is the transitivity of the event order and or the reverse relation generation. For example, considering A, B, and C are three events, if A came before B and B before C, then A came before C. And if A and B have a temporal relation then B and A have the reverse relation.

The datasets for temporal relation extraction differ in the relationships available, the type of entities that are linked, and when the relationship is considered or ignored. For instance, TCR (Ning, Feng, Wu, et al., 2019) considers only five types of temporal relationships: “before”, “includes”, “is included”, “simultaneously”, and “after”. While CAVEO (Chambers et al., 2014) has a relation named “vague” that relates unclear temporal relations between events. TimeBank (Pustejovsky et al., 2003) on the other hand also considers “ended by”, “During inv”, “Begun by”, “Begins”, “IBefore”, “IAfter”, “During”, and “Ends”.

Temporal relations have three types (Chambers, 2013): (1) relations between two events, (2) relations between events and time expressions, and (3) finally relations between time expressions and events. Some datasets added as annotations feature the event aspects label. In TimeML (Bethard, 2013) for example as “Progressive”, “Perfective”, “Perfective Progressive” or none. Time expressions are classified as Time, duration, date, or set. To identify the time value, time normalization is available such as TimeN time normalization system (Llorens et al., 2012). They match the tense of the verbs but don’t go beyond the culmination, point, process, and culmination process. Events are assigned polarities as positive and negative and their modularity such as “could”, “may”, “would” ... are also classified.

Multiple models such as NavyTime (Chambers, 2013), CAVEO (Chambers et al., 2014), UTTime (Laokulrat et al., 2013), ClearTK (Bethard, 2013), Sequential Models based on LSTM (Choubey & Huang, 2017), TE-KMN (Duan et al., 2018), or structured learning approach were proposed (Ning, Feng, & Roth, 2019). The models vary in the places they search for the relations between events. Some models only search for intra-sentences events, others go beyond. For instance, NavyTime searches for relations of events within the same sentence, events in adjacent sentences, and events in the same document. UTTime considers a relation between all events and documents creation time, events mentioned in the same sentence, events and time expression, and events in consecutive sentences. The model performance is diverse based on the datasets used and some models are built using multiple datasets. And evaluated their performances.

Since an effect occurs after its cause, causal analysis enables temporal inferences (Ning, Feng, Wu, et al., 2019). Some models considered working on both causal relationships and temporal relations and added constraints to learn causal relations based on event ordering. Other than the default previously mentioned constraints of transitivity and reverse inferences, the causal relation generates an ordering inference between the event and its cause. Some datasets are Causal-TimeBank (Mirza & Tonelli, 2016), EventStoryLine (Caselli & Vossen, 2017), or FinReason (P. Chen et al., 2021). Finally, TIMERS (Mathur et al., 2021) is a document-level approach that incorporates chain reasoning, causal prerequisite links, and future events to improve temporal relationship extraction.

In previous work, we introduced timeframes as frames of time that will contain multiple events that happened in a specific period of time. This approach aims to group events to be grouped together by detecting relationships between the different timeframes identified even though they are not mentioned in adjacent sentences. After detecting the relationship we will apply one of the related work approaches for the event ordering. In this paper, we will be using the CAEVO model based on the TimeBank-Dense dataset for its availability. In the next section, we will present the different types of timeframes identified, their use, and how to extract them.

Timeframes

As previously mentioned, event extraction and event ordering methods allow identifying multiple events in a text and creating temporal relationships with each other in case a relationship is mentioned. They also allow the extraction of numerous entities that are related to the events. If we focus on the events and temporal relations alone and aim to represent them in a graph where the events are the nodes and the temporal relationships are the edges, the result of a specific text will be mostly unconnected nodes. If a specific date is mentioned, it will be related to a specific event but events that are not connected to the dated event will remain without any temporal relation. And this result is the output of a single text. So if we want to go to a higher scall and analyze events from multiple texts, dealing with events that are not dated or have no time relation will be problematic to identify causal and impacts. We introduced in prior work three types of timeframe and how to extract them (Matta et al., 2022). In this paper, we will show how these timeframes can be used to group events by providing an innovative structural representation. In this section, we present the different timeframes before providing our proposed approach.

Publication Timeframes

The publication timeframe aims to situate a text in the time where it has been published by enabling a temporal index to the information extracted. From the analysis point of view, the state of the event mentioned in a text is directly related to this timeframe. By the state of the event we mean if it had been completed a long time ago or recently, if it's an ongoing event, if the event is about to start, if it will happen in the future, or if it had been canceled. The state of the events is relative to the publication date. For data extracted from news websites, we went through a cleaning phase in which we separate sentences that end with a point from other sentences. The sentences that don't end with a point are usually titles, references, or captions. The is a high chance of finding the publication date in those data. If not we consider the first three sentences and the last sentence to check for a specific pattern. The pattern that enables the identification of a publication timeframe is a date format alone or in case the date is semantically related to a verb the verb should be related to the publication vocabulary. For example, published, updated, issued….

Some textual data are already assigned a publication date such as posts on social media while being scrapped while some information doesn’t have a publishing date. For instance, data gathered on companies’ websites, such as product pages. Those pages are usually not used to talk about events but to describe companies and their goods. If the information extracted by nature of the text doesn't have a publication date we will assign a scrapped date to this information. This will allow us to compare the data scrapped from the same site over the years.

Spoken Timeframes

The spoken timeframe aims to isolate the parts of the text that are written directly the way they had been said. As distinguished in the literature, there are two types of speech; direct speech and reported speech. We are interested in the direct speech for temporality and verb tenses of the text inside the quotation are relative to the time it had actually been said. While the tense in the reported speech is relative and related to the text thus can directly be analyzed accordingly. We used pattern matching to identify sentences with direct speech. For this analysis, we replaced the text inside the quotation with an empty quotation to reduce the complexity of the dependency parsing. Then using a dependency parser, the first quotation must be the object of the verb in the sentence.

Spoken timeframes gather speech threads. In other words, if consecutive sentences had a direct speech in them they will be grouped together in a single spoken timeframe for we believe that they are related to each other. To each spoken timeframe identified we assign an identification key and the thread will be replaced by this key to keep track of its original place in the text. This will allow us, later on, to situate the speech in the narrative timeframe.

Narrative Timeframe

The narrative timeframe is a timeframe that groups multiple events mentioned in a text that are considered to have happened within a specific period of time. A text may have multiple narrative timeframes related to each other by temporal relationships. By default, a text is placed in a single narrative timeframe. Patterns based on temporal specifications in sentences had been defined to identify the presence of a new timeframe. It is very important to highlight what temporal relations generate the creation of a new narrative timeframe and the ones that don’t. We used a named entity recognition tool to identify the dates in the text. The model doesn’t only look for the date format but anything related to dates such as “Tomorrow”, “a few days ago” or “In 2022” but ignores temporal relations such as “before graduation” which will be considered later on as a relation between events. The position of the dates is also important; they must be at the beginning of the sentence. This positioning is major because it situates the following (the rest of the sentence, the paragraph, or the text) in the time. While the date mentioned in the middle or at the end of a sentence will only be considered as a relation to the events inside the sentence.

The tense of the verbs is also monitored. While going through the sentences to detect the previously mentioned patterns, we extract the tenses used in the current narrative timeframe. The verb tenses are classified as follows: anterior past, past, present, future, and anterior future. If the first sentence of the new timeframe has no common tenses with the previous one, we monitor the tenses with the following sentences because reusing the old tense may be considered a sign of going back to the prior narrative timeframe. We will settle with just generating a new timeframe and we will manage this issue while dealing with the relationship between the timeframes in the proposed approach in this paper. Note that the output of the narrative timeframe extraction is the distribution of the sentences of the text onto a single or multiple narrative timeframes.

For the timeframe extraction, the order goes as follows; we start with the publication timeframe extraction, then the spoken timeframe extraction, and finally the narrative timeframe extraction. Please note that the narrative timeframe will not have direct speech in them. The spoken timeframes are replaced in the narrative timeframes by their corresponding identifier that will enable the indexation. In the following section, we will present our approach in which we create relationships between timeframes and extract events and represent them in timeframes.

Timeframe-based Event Ordering

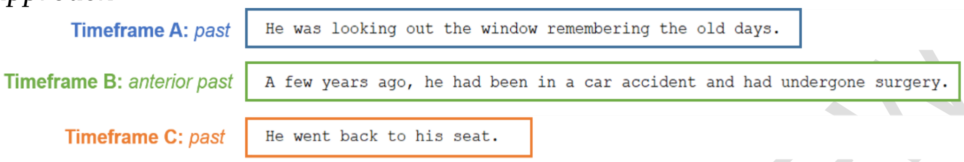
As previously presented, the timeframes extraction approach allows segmentation of a text onto publication, narrative, and spoken timeframes. Three steps are required for enabling the event ordering using those timeframes. The first step is generating relationships between the identified timeframes. For now, one type of relationship will be considered, the Narrative-Narrative timeframe. The second step will be ordering the events mentioned within a single timeframe. We will use the event ordering for both narrative and spoken timeframes but the relations between the spoken and narrative will be considered in future work. This section will be divided into two main parts that are consecutively the timeframe relation extraction and the event order. The last step enables the extraction of relations between time expressions and timeframes through a pattern-matching approach.

Timeframe Relation Extraction

For the extraction of Narrative to Narrative timeframe extraction we have two possible cases that triggered the creation of a new narrative timeframe. The first one is based on the use of time expression at the beginning of a sentence. If the time expression has an ordering element such as “ago” in “a few days ago”, the relation can be determined using the connective-lex; a lexicon for logical connectors in the text. We selected the English lexicon and specifically the temporal connectors. If a date is mentioned without a temporal connector, the date will be considered the temporal reference of that timeframe. The tenses with the previous timeframe will be compared to determine if a possible order can be generated.

The other trigger of narrative timeframe occurs when a narrative timeframe B is created having a tense that does not correspond to the previous timeframe A. The use of the tense in A might mean that we came back to the timeframe A which triggers the creation of a new timeframe C. "Figure 1" below shows a use case:

Figure 1. *Example of a use case highlighting the importance of the timeframe approach*



We can notice that the tense in the first sentence is "past", and the second sentence starting with "a few years ago" triggers the creation of a new narrative timeframe and has "anterior past" as a tense. Having "past" as tense in the third sentence may lead to the relation "after" with the timeframe A which is why our algorithm separates the timeframe B and C. Note that the tense we are using reflects the root verb tense in the sentence. In other words, if a sentence has multiple verbs within it, the main verb of the sentence will be considered. From this point, we are in the C timeframe and we are trying to find the temporal relation with the directly previous timeframe or the one before it. C is the only case that triggers the creation of a narrative timeframe that does not start with a time expression. We will use T_n, T_{n-1}, and T_{n-2} as the annotation of respectively the current timeframe, the previous one, and the one before it. The pattern goes as follows: if the last sentence of T_{n-1} has no tense in common with the first sentence of T_n and the tense of T_n is the same as the last sentence of T_{n-2}, T_n will be considered as happening "after" T_{n-2}.

Figure 2. *Temporal Relation Extraction between two Narrative Timeframes Algorithm*

```
Narative_Narative_Relation_Extraction(NarrativeTFList,NarrativeIDList,connector_lexicon):
oderedNarative = []
for i in range 1,length of NarrativeTFList, i++:
    timeX = Start_With_TimeX(NarrativeTFList[i]):
    if empty timeX:
        tn = NarrativeTFList[i]
        tn-1 = NarrativeTFList[i-1]
        tn-2 = NarrativeTFList[i-2]

        if not Check_Tense(tn,tn-1) and Check_Tense(tn,tn-2):
            add [NarrativeIDList[i],NarrativeIDList[i-2], 'after'] to oderedNarative
        esle:
            order = Get_Order_Tense(tn,tn-1)
            add [NarrativeIDList[i],NarrativeIDList[i-2], order] to oderedNarative
    else:
        date = Has_Date(timeX)
        if Not_Empty(date):
            add [NarrativeIDList[i-1],date[0], date[1]] to oderedNarative
        else:
            if Has_Connector(timeX):
                retation = Generate_Relation(timeX,connector_lexion)
                add [NarrativeIDList[i-1],NarrativeIDList[i], relation] to oderedNarative
            add [NarrativeIDList[i-1],NarrativeIDList[i], timeX] to oderedNarative
return oderedNarative
```

"Figure 2" presents the algorithm used for the Narrative to Narrative relation. It takes as input the list of narrative timeframe, a list of the corresponding identification key to the narrative timeframe, and the connector lexicon that will be used to generate the relationship in case a logical connector was used. The function returns a list of the timeframe identification couple with the relation that relates them. In the relation extraction, the first sentence is skipped since it doesn't have a prior timeframe to relate to. The algorithm uses depends on multiple functions:

1. The first one is Strat_With_TimeX which is a function that takes a timeframe as input and checks if the first noun phrase in the paragraph is a time expression. It returns the teme expression if found, else it returns an empty list.
2. The second function is Check_Tense which takes two timeframes and checks if the tense of the first sentence of the first argument and the last sentence of the second argument have any element in common.
3. The third function is Get_Order_Tense which takes two timeframes and generates an order between the two based on the tense used in the two timeframes.
4. Has_date is a function that takes a time expression as input and checks if the time expression has a date in it. The function will return a list in which the first element is the date and the second the connector related to the date. If no date were found the function will return an empty list.
5. Has_connector is a function that takes a time expression and checks the presence of a time connector in it.
6. Generate_Relation takes a timeframe starting with a time expression and the Connector_Lexicon which is the lexicon that will determine the relation implied by the time expression.

It is important to note that in case the time expression had a date, the timeframe will be related to the date. If the time expression did not have a date, a relationship with the time expression will relate the current timeframe to its previous one. If the time expression has a temporal connector it will trigger the creation of a default relationship such as 'before' or 'after'.

For the relation between Narrative timeframe and Spoken Timeframe, the spoken time frames will be analyzed independently and will have a "vague" temporal relationship with the narrative timeframe they appear in. In future work, we will elaborate patterns and rules for generating more specific relations between the event extracted from spoken timeframes and the narrative timeframe.

Event Extraction and Ordering

After extracting the timeframes, a system for event ordering and event extraction will be applied for each of the timeframes. The event extraction and ordering will be done by CAEVO, a model mentioned in the related work section. Note that any other model can be used and applied in our approach. CAEVO stands for CAscading EVent Ordering system since it uses the transitivity inference to provoke the generation of more event relations. CAEVO starts by extracting the events using NavyTime's word selection of events. Then it extracts the Time Expression using the SUTime, a library provided as part of the StandfordCoreNLP. It identifies and normalizes the time expression in texts. After extracting the events and the time expressions, the model extracts the temporal relation. The temporal relations are "b" for before, "i" for includes, "ii" for is included, "s" for simultaneously, "a" for after, and "v" for vague. The last functionality of the model is the generate relations using transitive inferences. CAEVO is trained using TimeBank-Dense (Cassidy et al., 2014), a newer version of TimeBank (Pustejovsky et al., 2003). Timebank is an annotated corpus for events, time expression, and the relation between them. TimeBank ignores states that are considered permanent or that will remain true regardless of the events that are happening in the text. TimeBank only searches for relationships within the same sentence and so TimeBank-Dense is a newer version that enables "denser" relation extraction that enables relations by creating relations between events in the same document. This model was selected for its disponibility essentially but can be replaced by any other event ordering system.

Time Expression and Timeframe Relation Extraction

We identified a specific type of time expression that is considered to connect the timeframe of occurrence to a different timeframe. Specific nouns and patterns were identified to enable this approach. "Table 1" provides the patterns and the relationships they created using them.

Table 1. *Identified patterns and the corresponding relation*

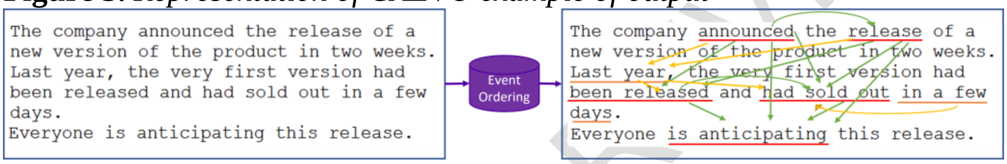
Patterns on Time Expressions	**Relation Created**
Yesterday	before
today	is_included
Tomorrow	after
<start_with> { this | these | last | next }	<corespnding> {is_included | is_included | before | after }
<ends> { ago }	before

The algorithm goes as follow, it considers the list of time expression extracted and try to find this patterns. If the time expression matches any pattern, we look for the timeframe it occurred in. If the timeframe is a narrative timeframe, a relationship between the time expression and the publication timeframe is created. If the time expression occurred in a spoken timeframe we must consider two cases. If the time expression is outside the quotation marks the relation is created with the publication timeframe, while if it was inside them then the relationship will be between the time expression and the narrative timeframe.

Results and Evaluation

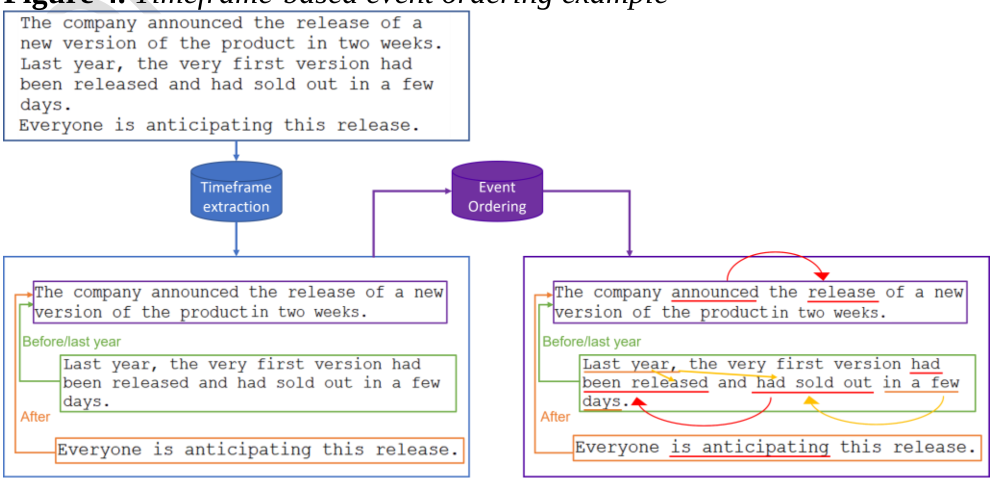
To highlight the added value of our approach we will present the output of our approach in contract with the output of the event extraction model. "Figure 3" presents the output of the CAEVO event ordering without the timeframe approach.

Figure 3. *Representation of CAEVO example of output*



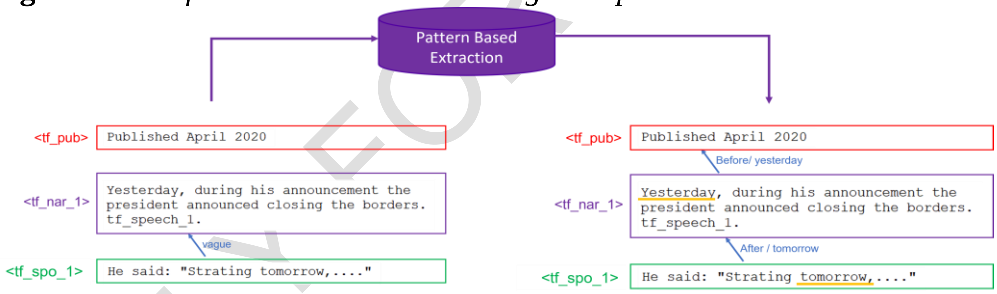
We can notice that within three simple sentences, the model detected 5 events and 2 time expressions, and generated 15 temporal relationships. 10 of these relationships were between events and 5 relations between the events and the time expression. Noting that the model used generates such results by using only the transition inference, the models using the inverse inference might end up with the double. For longer texts, the use of those relationships and their representation will result in a huge amount of relationships with a complexity of identifying the useful relations. While the models that don't use inferences and don't connect adjacent sentences events may end up ignoring important information as we can see in "Figure 3".

Figure 4. *Timeframe-based event ordering example*



"Figure 4" on the other hand shows the output of the same text but using the timeframe-based event ordering. We can notice the output of the timeframe extraction. The text was segmented into three narrative timeframes and those timeframes were connected via temporal relationships. The order is always present in the temporal relation between timeframes. We have before, after, includes, is included, simultaneously, and vague. When the model finds time expressions at the beginning of the timeframes, it associates them with the temporal relation extracted. After the timeframe extraction, the event ordering model will be applied to each of the timeframes. We can notice that the same events were identified but the events that belong to different timeframes were not connected. Using our approach, fewer relations were created but all the semantic relations that the event ordering model had been present. We can notice in "Figure 4" how the timeframe in purple is connected with the other two timeframes. The green timeframe happened before the purple while the second happened after it. This resulted in irrelevance in creating relations between events present in two different timeframes. For example, there was no need in creating a relation between "announced" and "had been released" or "is anticipating". This relation is provided by the relations between the time timeframes which will be inherited. In the default model, those relations were created based on inferences. Three relations were vague but our model had no vague relation detected in this example.

Figure 5. *Timeframe-based event ordering example*



"Figure 5" shows the output of the final step of our proposed approach. It took three timeframes for a single text that had already been analyzed. We only represented the temporal relation between the timeframes. We can notice the "vague" relation provided by the first step of our model. After the pattern matching phase, two time expressions matched our patterns. The first one was in the narrative timeframe which led to the generation of a relation with the publication timeframe. The second one was in the quotation marks of the spoken timeframe which led to the generation of a relation with the narrative timeframe in which the spoken timeframe occurred.

We considered 20 news articles extracted from four news websites: CNN[1], BBC[2], France24[3], GlobalNews[4]. The news articles extracted were from

[1]https://edition.cnn.com/
[2]https://www.bbc.com/news
[3]https://www.france24.com/en/

three different domains: politics, industry, and global news. Note that the use of such a small amount of data was to facilitate the manual evaluation of the output of the approach since we are still in the early stages of the timeframe approach. To compare the output of our timeframe approach and the CAEVO approach, the number of events extracted was the same. It was evident that since the same model was extracting the events, we had the same amount of events extracted. The number of relationships was reduced by an average of 64%.

Table 2. *Comparison of CAEVO and Timeframe-Based Approach using CAEVO*

	Relation Created	**Timeframe-based Approach**
Average number of events with the time expression	79.4	79.4
Average number of temporal relation	184.5	66.4
Most frequent relation	vague (100%)	none_vague (95%) vague (5%)

We noticed the dominance of the temporal relation “vague” in news articles used in all of the event ordering provided by CAEVO. As for the most frequent relation using our approach the “none-vague” relations were generally more frequent. To be more precise the most frequent relations were Before and after combined in the none-vague relation. We considered the relationship type and their inverse as the same type of relation since by inferences we can generate their inverse. A single article showed “vague” as the most frequent relation, this text had only one spoken timeframe and a single narrative timeframe. As for the rest, most articles had at least 3 to 4 narrative or spoken timeframes which reduced considerably the number of relationships.

Conclusions

Event extraction and event ordering are two essential fields in Information Extraction and Knowledge Discovery from textual data. In this paper, we presented a timeframe-based event ordering approach that is compatible with any event ordering models available. We used the CAEVO event ordering system and applied it to the timeframes extracted. The timeframe approach enabled a segmentation of the text on multiple narrative timeframes and spoken timeframes that were the input of the event ordering model. We provided methods to extract temporal relationships between multiple narrative timeframes and how the spoken timeframes will be handled for the time being.

[4]https://globalnews.ca/

This approach is a starting point for an event evolution identification approach. Our next phase will be focused on another field in Event Extraction called Event Disambiguation. It's a branch that focuses on identifying the occurrence of the same event multiple times in a single text or in multiple texts and enriching the arguments extracted related to that event. When identifying the mention of the same event in multiple places in a single text or in multiple texts, we can compare the multiple states of the event and generate an evolution detection. If we are dealing with multiple texts, the publication timeframe will play a role in situating the state of a specific event mentioned and analyzing its evolution throughout time.

References

Bachimont, B., Isaac, A., & Troncy, R. (2002). Semantic Commitment for Designing Ontologies: A Proposal. In A. Gómez-Pérez & V. R. Benjamins (Eds.), *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web* (Vol. 2473, pp. 114–121). Springer Berlin Heidelberg.

Bethard, S. (2013). ClearTK-TimeML: A minimalist approach to TempEval 2013. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 10–14.

Caselli, T., & Vossen, P. (2017). The Event StoryLine Corpus: A New Benchmark for Causal and Temporal Relation Extraction. *Proceedings of the Events and Stories in the News Workshop*, 77–86.

Cassidy, T., McDowell, B., Chambers, N., & Bethard, S. (2014). *An Annotation Framework for Dense Event Ordering*. CARNEGIE-MELLON UNIV PITTSBURGH PA.

Chambers, N. (2013). NavyTime: Event and Time Ordering from Raw Text. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 73–77.

Chambers, N., Cassidy, T., McDowell, B., & Bethard, S. (2014). Dense Event Ordering with a Multi-Pass Architecture. *Transactions of the Association for Computational Linguistics*, *2*, 273–284.

Chen, P., Liu, K., Chen, Y., Wang, T., & Zhao, J. (2021). Probing into the Root: A Dataset for Reason Extraction of Structural Events from Financial Documents. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2042–2048.

Chen, Y., Ding, Z., Zheng, Q., Qin, Y., Huang, R., & Shah, N. (2020). A History and Theory of Textual Event Detection and Recognition. *IEEE Access*, *8*, 201371–201392.

Choubey, P. K., & Huang, R. (2017). A Sequential Model for Classifying Temporal Relations between Intra-Sentence Events.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Duan, X., Tang, S., Zhang, S., Zhang, Y., Zhao, Z., Xue, J., Zhuang, Y., & Wu, F. (2018). Temporality-enhanced knowledgememory network for factoid question answering. *Frontiers of Information Technology & Electronic Engineering*, *19*(1), 104–115.

Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge Discovery in Databases: An Overview. *AI Magazine*, *13*(3), 57–57.

Hogenboom, F., Frasincar, F., Kaymak, U., de Jong, F., & Caron, E. (2016). A Survey of event extraction methods from text for decision support systems. *Decision Support Systems*, *85*, 12–22.

Hu, D. (2020). An Introductory Survey on Attention Mechanisms in NLP Problems. In Y. Bi, R. Bhatia, & S. Kapoor (Eds.), *Intelligent Systems and Applications* (pp. 432–448). Springer International Publishing.

Jacobs, G., & Hoste, V. (2022). SENTiVENT: Enabling supervised information extraction of company-specific events in economic and financial news. *Language Resources and Evaluation*, *56*(1), 225–257.

Johansen, T. K. (2021). *Productive Knowledge in Ancient Philosophy: The Concept of Technê*. Cambridge University Press.

Laokulrat, N., Miwa, M., Tsuruoka, Y., & Chikayama, T. (2013). UTTime: Temporal Relation Classification using Deep Syntactic Features. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 88–92.

Leeuwenberg, A., & Moens, M.-F. (2017). Structured Learning for Temporal Relation Extraction from Clinical Records. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 1150–1158.

Li, Q., Li, J., Sheng, J., Cui, S., Wu, J., Hei, Y., Peng, H., Guo, S., Wang, L., Beheshti, A., & Yu, P. S. (2021). A Compact Survey on Event Extraction: Approaches and Applications.

Li, Q., Peng, H., Li, J., Wu, J., Ning, Y., Wang, L., Yu, P. S., & Wang, Z. (2022). Reinforcement Learning-Based Dialogue Guided Event Extraction to Exploit Argument Relations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *30*, 520–533.

Lim, C.-G., Jeong, Y.-S., & Choi, H.-J. (2019). Survey of Temporal Information Extraction. *Journal of Information Processing Systems*, *15*(4), 931–956.

Liu, J., Min, L., & Huang, X. (2021). An overview of event extraction and its applications.

Liu, K., Chen, Y., Liu, J., Zuo, X., & Zhao, J. (2020). Extracting Events and Their Relations from Texts: A Survey on Recent Research Progress and Challenges. *AI Open*, *1*, 22–39.

Llorens, H., Derczynski, L., Gaizauskas, R., & Saquete, E. (2012). TIMEN: An Open Temporal Expression Normalisation Resource. *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, 3044–3051.

Mathur, P., Jain, R., Dernoncourt, F., Morariu, V., Tran, Q. H., & Manocha, D. (2021). TIMERS: Document-level Temporal Relation Extraction. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 524–533.

Matta No., Matta N., Giret E., & Declercq N. (2021), Enhancing Textual Knowledge Discovery using a Context-Awareness Approach. *The 8th Annual Conf. on Computational Science & Computational Intelligence (CSCI'21)*, in press.

Matta No., Matta N., Declercq N., & Marcante A. (2022), Semantic Patterns to structure TimeFrames in Text. *Semantic Patterns to structure TimeFrames in Text (INTELLI 2022),* in press.

Mejri, M., & Akaichi, J. (2017). A Survey of Textual Event Extraction from Social Networks. *LPKM*.

Mirza, P., & Tonelli, S. (2016). *CATENA: CAusal and TEmporal relation extraction from NAtural language texts*. 12.

Moens, M., & Steedman, M. (1988). Temporal Ontology and Temporal Reference. *Computational Linguistics*, *14*(2), 15–28.

Ning, Q., Feng, Z., & Roth, D. (2019). A Structured Learning Approach to Temporal Relation Extraction.

Ning, Q., Feng, Z., Wu, H., & Roth, D. (2019). Joint Reasoning for Temporal and Causal Relations.

Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., & Lazo, M. (2003). The TimeBank corpus. *Proceedings of Corpus Linguistics*.

Xiang, W., & Wang, B. (2019). A Survey of Event Extraction From Text. *IEEE Access*, *7*, 173111–173137.

Zhou, H., Yin, H., Zheng, H., & Li, Y. (2020). A survey on multi-modal social event detection. *Knowledge-Based Systems*, *195*, 105695.