Strings for Temporal Annotation and Semantic Representation of Events

by

David Woods

A dissertation submitted

in fulfillment of the requirements

for the Degree of

Doctor of Philosophy

University of Dublin, Trinity College

December 2020

Supervised by: Dr Tim Fernando, Dr Carl Vogel

Declaration

I, the undersigned, declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I, the undersigned, agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

David Woods

December 2020

Abstract

Acknowledgements

I would like to thank my parents, Margaret and Graham, who never doubted I would do my best; my brother, Fergus, often a much-needed reminder of normal life; Conor and Katie, who inspired me to start; Adelais, who encouraged me to finish; and all the members of DU Trampoline Club, who gave me a reason to stick around. I hope it was worth it!

My thanks go also to my supervisors: Tim, for being patient with me even when I was struggling, and Carl, who often managed to reassure me that I wasn't going down completely the wrong path.

This research is supported by Science Foundation Ireland (SFI) through the CNGL Programme (Grant 12/CE/I2267) in the ADAPT Centre (`https://www.adaptcentre.ie`) at Trinity College Dublin. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

Related Publications

During the course of my studies, I was an author on three papers that were accepted for publication, listed below.

- "Towards Efficient String Processing of Annotated Events" (Woods et al., 2017), describing the use of strings to model temporal data such as could be found in text annotated with ISO-TimeML. Presented at the 13th Joint ISO-ACL Workshop on Interoperable Semantic Annotation in Montpellier, France.

- "Improving String Processing for Temporal Relations" (Woods and Fernando, 2018), discussing refinements to the previously described string-based model, such as varied granularity. Presented at the 14th Joint ISO-ACL Workshop on Interoperable Semantic Annotation, colocated with COLING 2018 in Santa Fé, New Mexico, USA.

- "MSO with tests and reducts" (Fernando et al., 2019), discussing differing string granularities in the context of tests within Monadic Second Order logic. Presented at the 14th International Conference on Finite-State Methods and Natural Language Processing in Dresden, Germany.

Contents

1 Introduction

This thesis will explore and describe the use of strings as models to represent temporal information—data concerning times and events—for use in computational systems which deal with knowledge-based reasoning in some way. Such systems rely on temporal information in order to perform accurately. For example, a question-answering system that is asked "Will it rain next week?" must be able to locate itself temporally such that it knows what the current time is, the relation between that time and the queried time, and whether raining events have been forecast during the period that it defines using the phrase "next week".

2 Relevant Literature

In this chapter, the existing literature related to the major topics of the thesis are reviewed and analysed. A gap is identified, which the remaineder of this work seeks to fill.

2.1 Times and Events

The concept of time has fascinated researchers for millenia, and as such, a great deal of work exists on the topic. What follows here focuses on the formal study of temporality in language.

2.1.1 Allen Relations

The seminal work of James F. Allen's *Maintaining Knowledge about Temporal Intervals* has pushed the field since its publication in 1983, and indeed the framework he described drives some of the design decisions in the present work.

2.1.2 Tense and Aspect

Reichenbach's (1947) theory of tense and aspect allows for the temporal placement of an event time in relation to a speech time and a reference time. The relative orderings of these three times gives rise to the categorisations of tense and aspect. [NB[1]]

2.2 Temporal Annotation

Ideally, a human using an artificially intelligent system won't have to consider annotation, except in the case when that itself is the goal. In any case, for now annotated text is a crucial source of data for working with models such as are presented in this work. When

[1] Bring up TEA here. Also worth mentioning Vendler and Dowty in relation to lexically telic/atelic verbs, which inform the stative-based approach of the strings and thus block-compression. Tim mentioned that this is a difference from Schwer's S-Words.

it comes to temporal annotation, the international standard is TimeML, specifically ISO-TimeML. [NB[2]]

2.2.1 ISO-TimeML

TimeML (Pustejovsky et al., 2005) was initially designed with the goal of improving question-answering systems by marking up texts so as to give events explicit temporal locations.

TimeML vs ISO-TimeML: the ISO version is the standard. However, the TimeBank corpus uses the older version.

2.2.2 TimeBank

The TimeBank corpus (Pustejovsky et al., 2006) collects news articles that were manually annotated with TimeML. While it does contain some inconsistencies as a result of human error, it remains one of the largest sources for documents marked up with TimeML.

2.3 Temporal Semantics

The following sections describe some of the existing ways that the semantics of times and events can be represented. [NB[3]]

2.3.1 Discourse Representation Theory

Kamp and Reyle (1993) introduced Discourse Representation Theory (DRT) as a framework for semantically representing information derived from a text. DRT separates discourse referents (the entities under discussion) from Discourse Representation Structure (DRS) conditions, which describe what is known about the referents.

[2] Also talk about Tango and Verhagen's T-BOX.
[3] This section needs more—Prior's TL, FOL, SOL, Event Calculus?, Frames?, FST?

[NB[4]]

2.3.2 Boxer

Automatic parsing of text into DRSs is a popular task [NB[5]], and Bos (2008) released the first version of the Boxer software which claimed to do this with over 95% coverage for semantic analysis of newswire texts, though it was noted in the release that performance for temporal data was not as strong as its other areas.

There is a newer version of Boxer implemented as part of the Parallel Meaning Bank [Citation needed!] toolchain—however, this version had unfortunately not been made available for general use at the time of writing.

[4] Harry Bunt's slides from his DCLRS talk would be useful here, else ref "A semantic annotation scheme for quantication" (Bunt, 2019)

[5] Find the shared task from 2019—I'm sure it was on the PMB website, but I can't find it at present

3 Finite-State Temporality

Finite-state methods may be applied to temporal semantics in an approach known as *finite-state temporality* (Fernando and Nairn, 2005) [NB[6]]. In this chapter, strings are demonstrated as a tool of choice in modelling sequences of times and events for use within this approach. The reasoning behind the creation of such temporal strings is shown, along with their mechanics, and a discussion on the granularity of temporal information, i.e. how much "zoom" is useful when looking at an event.

A number of operations are described for working with these strings, in particular *superposition* for combining the data from multiple strings. These manipulations will prove useful when reasoning about event relations, and also in maximising data density, as will be seen.

The strings are then shown as applied to creating a functional timeline from the temporal information in a piece of text, and also how they may be useful elsewhere, such as in the area of scheduling, using a variation of the Zebra Puzzle [Citation needed!].

3.1 Strings for Times and Events

A string is a basic computational entity, defined as a finite sequence of symbols selected from some finite alphabet A. They are amenable to manipulation using finite-state methods, something lacking in the infinite models of predicate logic.

How should events be labelled? Some say an event is defined in part by its participants [Citation needed!]. Many of the procedures defined in this work which use these strings operate primarily on a syntactic level rather than depending on the lexical semantics of individual words, in order to produce an approach that works broadly. As such, in most

[6]Probably need to go into more detail on what FST is as defined here. Definitely need to mention why finite-state as a whole is good. Maybe mention MSO in this opening bit?

cases, simple labels suffice for the purposes of describing the mechanics used here—this follows TimeML's standard of using labels such as `"e1"` or `"t2"` for events and times. However, it will be useful to keep reference to a fuller picture of the data when possible, particularly when drawing inferences (see 3.2.2 and 4.3.2).

3.1.1 Creating Strings

In order to create a string, first fix a set A of symbols which represent the times and events under discussion.

Sets of strings are languages [NB[7]].

3.1.2 Granularity: Points vs Intervals vs Semi-intervals

A decision must be made in regards to what is regarded as 'primitive' in a string, in terms of whether a basic component is considered as instantaneous, or as having some duration. In the first view, only points exist, and so an event will have a beginning and an ending (left and right borders), each of which represented by a symbol in the string. In the second, entire intervals are used instead of points. Two points may only be related in three ways: $<$, $=$, $>$. Two intervals, however, may be related in thirteen different ways, precisely, the Allen Relations [Citation needed!].

There are advantages to each approach: ... e.g. intervals are intuitive and infinitely divisible and allow for more specific relation-labels; points are simpler and incomplete information doesn't break the system by forcing disjunctions.

A third option exists: semi-intervals, as described by Freksa (1992). By treating the borders of an event as intervals, this expands the number of available relations between two events to 31.

[7] Maybe mention this in 3.1 rather than here? See how it goes

Ultimately, intervals are chosen for a number of reasons: primarily because ISO-TimeML, the international standard for temporal annotation, uses intervals for its TLINKs, but also they seem the most intuitive [NB[8]]. Worth noting that there are translations available between the different granularities.

3.1.3 String Operations

Many many operations. The most important is **superposition** (and it's more advanced forms)! Also we have block-compression, reduct, vocabulary, projection, border translations. Note also that there are equivalent operations for strings which use points instead of intervals.

Discuss the language level operations here too.

3.2 Applications

The following will outline some of the ways that the use of string-based FST can applied to problems in NLP.

3.2.1 Timelines from Texts

Strings, as entities comprised of sequential components, have an intuitive comparison to a traditionally linear view of time. That is, that events which have not yet occurred are ahead of us, and events which occurred in the past are behind us—though not all languages or cultures perceive time in this way, it is common cross-linguistically to use some spatial reference points when discussing temporality [Citation needed! – see Sec 1 para 1 of: `https://onlinelibrary.wiley.com/doi/full/10.1111/cogs.12804`]. Regardless of spatial orientation, this perception of time maps well to a sequential representation

[8] This is vague and subjective; defend it better

as is found with strings. Perceived directionality of time is not an issue, as it is a small modification to change the direction of a string.

A lot of information can be derived from a text document, depending on what one is looking for. Here, the focus is on isolating the events and times that are mentioned or implied in the text. By extracting this information, a timeline is built which gives a picture of the content of the document, which may reveal insights not obvious when looking at the text as a whole.

3.2.2 Inferring New Information

A text which has been manually annotated for temporal inforamation will typically mark up the most important relations between events, at least as the annotator saw them. By extracting and making plain the timeline, any inconsistencies are immediately revealed, whether these are the fault of a problematic document or human error. Further, relations between times and events which were not previously obvious can be spelled out clearly.

This can be extended over multiple documents by keeping track of the semantics of the elements in a string—being able to map between the symbols and their meanings—and linking documents which refer to the same events. This can provide a way for checking consistency between reports which originate from different sources, or a way for augmenting an existing timeline with new data.

Additionally, conclusions can be drawn about whether certain inferences can be made based on existing data by determining the gap between premise strings and strings representing the question statement. That is, deriving the information that could be added to the premises to make the conclusion consistent with the known information (see 4.3.3).

3.2.3 Scheduling (Zebra Puzzle)

Scheduling is a problem. The Zebra/Einstein Puzzle is another problem. They may seem dissimilar, but in fact, there are a number of parallels. Both problems can be viewed as constraint satisfaction problems. Although the Zebra Puzzle concerns physical constraints, these can be modelled as sequences for which one can use strings to represent. If a "house" is conceptualised as a box in a string i.e. a set, the elements contained within are the properties of that house according to the puzzle. For example, $\{red, english, zebra, oj, kools\}$. The left/right spatial relations are thought of in the same way as the previous/next boxes in a string.

Below is presented a variant of the puzzle using clues pertaining to temporal relations instead of spatial ones.

4 Methods

This chapter details the approaches that were taken to produce the results.

4.1 Extracting Strings from Annotated Text

The primary source of data is the TimeBank corpus which is one of the largest available collections of documents which are annotated with TimeML. Events are marked up with an <EVENT> tag, and times with the <TIMEX3> tag. Within TimeML, they are treated as intervals (see 3.1.2) which becomes relevant to the present work in the way they are related.

4.1.1 TLINKs

TimeML primarily uses <TLINK> tags to annotate relations between marked up events and times. These serve as the basis for initial string creation, as they provide the two intervals that are being related, as well as the relation between them. The relation set used is specific to TimeML, but has its roots in the set of Allen Relations—see table [Figure reference needed!] below.

A translation is built from <TLINK> to string, using the tag's attributes to provide the data, and the Allen Relations as an intermediary step, as per table [Figure reference needed!]. For example: [NB[9]]. Any events and times that feature in the document but are not mentioned in a <TLINK> are kept aside for now, as there is not a given connection from these events to the rest of the timeline.

After creating strings from the <TLINK>s, the next step is to start building the timeline, using the superposition technique described in 3.1.3. This allows for connections to

[9] example here please

be built between events and times that were not explicitly related by the annotation. If every time and event in the text is related to every other time and event, there is *temporal closure* in the text (see 2.1.1 [NB[10]]). For there to be temporal closure in a TimeML document would be a large amount of work for an annotator, as for N events/times there would be $N(N-1)/2$ relations between them, which increases quickly: at $N = 10$ there are 45 relations, at $N = 50$ there are 1225 relations. This becomes unwieldy for a human to annotate, especially given that it is generally understood by the human reader that if A precedes B, and B precedes C, then A also precedes C, so it feels unnecessary to include this latter relation. However, an automated system does not have this inherent knowledge, and needs a way to calculate the implied relations.

The various relations can be combined according to the table in Allen (1983) [Citation needed! – get the page and figure number], reproduced in 2.1.1 [Figure reference needed!], and again below using strings in table [Figure reference needed!]. The advantage to using strings is that the extension beyond three events is simple [NB[11]]. It should be noted that several combinations of relations produce a disjunction of relations, which follows from the fact that these combinations of $a \cdot b$ and $b \cdot c$ don't provide enough information to determine the exact relation $a \cdot c$. It may be possible to narrow this down, though, with the addition of further relations connecting other intervals $(d, e, f, ...)$ with some or all of a, b, c.

4.1.2 Handling Incomplete Data

When dealing with any temporal information, there is often a lack of specificity that means temporal closure cannot be calculated. For example, knowing that event a occurred before

[10] See transfer p6 – maybe include the graph of $n(n-1)/2, n \geq 2$?
[11] Defend this – I think there is something in ISA13 paper – or remove

event c $\boxed{}\boxed{a}\boxed{}\boxed{c}\boxed{}$ and that event b also occurred before c $\boxed{}\boxed{b}\boxed{}\boxed{c}\boxed{}$ does not impart any information on the relation between a and b. If this is all of the available data, then it is only possible to choose this relation $a \bullet b$ with a precision of $\frac{1}{13}$ chance. However, further information which includes other relations for each of a and b individually may eventually reveal their connection.

See also, using semi-intervals to simplify disjunctions of Allen Relations. Minimal sets of maximal strings.

Freksa relations using semi-intervals: the appearance of $\alpha(a)$ within a box represents a negation of the fluent a conjoined with a formula stating that a will be true in a subsequent box; similarly, $\omega(a)$ represents a negation of the fluent a conjoined with a formula stating that a was true in a previous box.

(1) $$\alpha_a(x) := \exists y(x < y \wedge y \in P_a \wedge \forall z(z < y \rightarrow z \notin P_a))$$

(2) $$\omega_a(x) := \exists y(y < x \wedge y \in P_a \wedge \forall z(y < z \rightarrow z \notin P_a))$$

Note that it is convenient to write $\alpha(a)$ for $\alpha_a(x)$ when using the box-notation.

A string may be translated to one using semi-intervals by placing $\alpha(f)$ in every box preceding one in which a fluent f appears, and $\omega(f)$ in every box succeeding it, repeating this for each $f \in A$. Thus a string $s = \boxed{}\boxed{a}\boxed{b}\boxed{}$ becomes $si(s) = \boxed{\alpha(a), \alpha(b)}\boxed{\alpha(b)}\boxed{\omega(a)}\boxed{\omega(a), \omega(b)}$, where each of the fluents which originally appeared in the string have also been removed. It should be noticed that strings which use semi-intervals do not (necessarily) feature empty boxes at each end. This is due to the fact that a semi-interval need not be finitely bounded. In fact

This mechanism allows for partially known information to be represented using strings.

For example, the string $\boxed{\alpha(a), \alpha(b)}\,\boxed{}$ represents the knowledge that the events labelled a and b both begin at the same moment, without stating anything about when they each finish—they may end simultaneously, a may finish before b, or b may finish before a. Which of these states is true is unknown without further data.

Semi-intervals allow for representing a greater range of event relations than the Allen relations do alone. Freksa [Citation needed!] describes 18 new relations which are made up of disjunctions of Allen's relations, based on the concept of cognitive neighbourhoods. That is, groupings of relations that make intuitive sense and can be derived from compositions of Allen relations [NB[12]] [NB[13]].

Many of the Freksa relations can be captured using the semi-intervals [NB[14]] in the box-notation.

For example, the relation *older* corresponds with a disjunction of five Allen relations: *before*, *meets*, *overlaps*, *contains*, and *finished by*. The similarities between these five

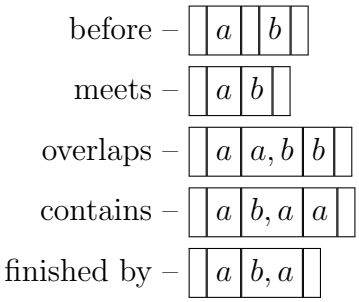

Figure 1: The Freksa relation *older*

relations becomes more apparent when they are translated to use semi-intervals instead:

Each of these five strings projects (using a block-compressed reduct) to the string $\boxed{\alpha(a), \alpha(b)}\,\boxed{\alpha(b)}\,\boxed{}$, meaning the relation a *older* b can be represented using just this one string, instead of five. This does raise the question, however, of whether the tradeoffs

[12] expand, clarify "intuitive", decide between Allen/Allen's, try to use less "relations"?
[13] include table or graphic of Freksa's relations, eg p18 or p21
[14] duh, he designed them that way

Figure 2: The Freksa relation *older* using semi-intervals, before block-compressed reduct

are worth it: a great reduction in the cardinality of the timeline set can be achieved, but at the cost of using a more complex vocabulary and reducing the precision of the known information.

Unfortunately, only 10 of the 18 Freksa relations can be described using a single string without further complicating the vocabulary which may appear within a box in a string. Since all of these new relations are disjunctions of Allen relations, it follows that it may be acceptable to include disjunctions of semi-intervals inside a box, such as $\boxed{\alpha(a) \vee \alpha(b)}$, which is interpreted as one might expect: either $\alpha(a)$ appears in the box, or $\alpha(b)$ does, or they both do. This allowance admits a further 5 Freksa relations to be described in single strings. Of the remaining 3 relations, the *unknown* relation may also be described using a single string, but it is trivial, since it encompasses a disjunction of all 13 Allen relations, and is formed by a simple disjunction of all possible semi-inteval symbols: $\boxed{\alpha(a) \vee \alpha(b) \vee \omega(a) \vee \omega(b) \vee \epsilon}$.

Below is a table of the 18 Freksa relations, the Allen relations they comprise, and the string which they will project to. [NB[15]]

The last two relations on this list cannot be represented using a single string, and are instead must use conjunctions of pairs of strings:

[15] tidy up ordering of allens

Freksa	Allen	string
unknown	b, bi, m, mi, s, si, f, fi, d, di, o, oi, e	$\alpha(a) \vee \alpha(b) \vee \omega(a) \vee \omega(b) \vee \epsilon$
older	b, m, o, di, fi	$\alpha(a), \alpha(b) \mid \alpha(b)$
younger	bi, mi, oi, d, f	$\alpha(a), \alpha(b) \mid \alpha(a)$
head to head	s, si, e	$\alpha(a), \alpha(b)$
tail to tail	f, fi, e	$\omega(a), \omega(b)$
survived by	b, m, s, d, o	$\omega(a) \mid \omega(a), \omega(b)$
survives	bi, mi, si, di, oi	$\omega(b) \mid \omega(a), \omega(b)$
born before death	b, m, s, si, f, fi, d, di, o, oi, e	$\alpha(a) \mid \omega(b)$
died after birth	bi, mi, s, si, f, fi, d, di, o, oi, e	$\alpha(b) \mid \omega(a)$
precedes	b, m	$\alpha(b) \vee \omega(a)$
succeeds	bi, mi	$\alpha(a) \vee \omega(b)$
contemporary	s, si, f, fi, d, di, o, oi, e	$\alpha(a) \vee \alpha(b) \mid \omega(a) \vee \omega(b)$
older contemporary	o, fi, di	$\alpha(a), \alpha(b) \mid \alpha(b) \mid \omega(a) \vee \omega(b)$
younger contemporary	oi, f, d	$\alpha(a), \alpha(b) \mid \alpha(a) \mid \omega(a) \vee \omega(b)$
surviving contemporary	di, si, oi	$\alpha(a) \mid \omega(b) \mid \omega(a), \omega(b)$
survived by contemporary	d, s, o	$\alpha(b) \mid \omega(a) \mid \omega(a), \omega(b)$
older and survived by	b, m, o	...
younger and survives	bi, mi, oi	...

Figure 3: The Freksa relations and the strings they project to

Freksa	Allen	strings
older and survived by	b, m, o	$\alpha(a), \alpha(b) \mid \alpha(b) \mid \;\wedge\; \omega(a) \mid \omega(a), \omega(b)$
younger and survives	bi, mi, oi	$\alpha(a), \alpha(b) \mid \alpha(a) \mid \;\wedge\; \omega(b) \mid \omega(a), \omega(b)$

Figure 4: The remaining Freksa relations and the strings they project to

4.2 Enhancing Strings using DRT

Automatic DRS-parsing of plain text is a task which has recently seen new approaches [Citation needed! – shared task, see fsmnlp paper sec 1]. It is possible to leverage the temporal information that features in a DRS to create strings. While the relations generally found using this approach are less specific than those found in TimeML [NB[16]], the advantage of using DRSs is that elements such as event participants are described.

Boxer (Bos, 2008) is one available tool for parsing a text into one or more DRSs, though the temporal information in this version struggles a bit. A newer version of the

[16] compare the relations

tool is used as part of the Parallel Meaning Bank [Citation needed!]toolchain, although it has not been made publicly available at present.

4.2.1 Parallel Meaning Bank

Under the assumption that the version of Boxer that is used in PMB would become available at some point, here is a description of PMB, and how the present work utilises the data within it as a demonstration of using text that has been parsed into DRSs can be transformed into strings for temporal reasoning.

Discourse relations are also a thing here.

4.2.2 VerbNet and WordNet

Two resources which are used in PMB and that can be leveraged are VerbNet, which supplies the semantic roles in a DRS, and WordNet, which is used to specify the particular sense of a verb (or other word). The version of Boxer used in the PMB includes this information in its output.

This data can be useful when trying to make links between events which have not been given a specific relation. For example, finding the closest hypernym between two verbs, or checking verbs which share participants may help to inform the building of relations (see 4.3.2)

4.3 Reasoning with Strings

4.3.1 Superposition

4.3.2 Event Ontology

4.3.3 Residuals and Gaps

5 Implementation

5.1 Back-end Pipeline

5.2 Front-end Interface

6 Evaluation

6.1 Timeline Validity

6.2 FRACAS Semantic Test Suite

6.3 Correctness of Code

7 Conclusion

Bibliography

Allen, J. F. (1983). Maintaining Knowledge About Temporal Intervals. *Communications of the ACM*, 26(11):832–843.

Bos, J. (2008). Wide-Coverage Semantic Analysis with Boxer. In *Proceedings of the 2008 Conference on Semantics in Text Processing - STEP '08*, pages 277–286.

Fernando, T. and Nairn, R. (2005). Entailments in finite-state temporality. In *Proceedings of the Sixth International Workshop on Computational Semantics*, pages 128–138, Tilburg University.

Fernando, T., Woods, D., and Vogel, C. (2019). Mso with tests and reducts. In *Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing*, pages 27–36.

Freksa, C. (1992). Temporal Reasoning Based on Semi-Intervals. *Artificial Intelligence*, 54:199–227.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Dordrecht: Kluwer.

Pustejovsky, J., Knippen, R., Littman, J., and Saurí, R. (2005). Temporal and Event Information in Natural Language Text. *Language Resources and Evaluation*, 39(2-3):123–164.

Pustejovsky, J., Verhagen, M., Sauri, R., Littman, J., Gaizauskas, R., Katz, G., Mani, I., Knippen, R., and Setzer, A. (2006). TimeBank 1.2 LDC2006T08. Web Download: `https://catalog.ldc.upenn.edu/LDC2006T08`. Philadelphia: Linguistic Data Consortium.

Reichenbach, H. (1947). *Elements of symbolic logic*. New York: Macmillan.

Woods, D. and Fernando, T. (2018). Improving String Processing for Temporal Relations. In *Proceedings of the 14th Joint ISO-ACL Workshop on Interoperable Semantic Annotation (ISA-14)*, pages 76–86.

Woods, D., Fernando, T., and Vogel, C. (2017). Towards Efficient String Processing of Annotated Events. In *Proceedings of the 13th Joint ISO-ACL Workshop on Interoperable Semantic Annotation (ISA-13)*, pages 124–133.

Appendices

Python Code