Simplifying annotation of intersections in time normalization annotation: exploring syntactic and semantic validation

Peiwen Su and **Steven Bethard**
University of Arizona
Tucson, AZ, USA
{peiwensu,bethard}@email.arizona.edu

Abstract

While annotating normalized times in food security documents, we found that the semantically compositional annotation for time normalization (SCATE) scheme required several near-duplicate annotations to get the correct semantics for expressions like *Nov. 7th to 11th 2021*. To reduce this problem, we explored replacing SCATE's SUB-INTERVAL property with a SUPER-INTERVAL property, that is, making the smallest units (e.g., *7th* and *11th*) rather than the largest units (e.g., *2021*) the heads of the intersection chains. To ensure that the semantics of annotated time intervals remained unaltered despite our changes to the syntax of the annotation scheme, we applied several different techniques to validate our changes. These validation techniques detected and allowed us to resolve several important bugs in our automated translation from SUB-INTERVAL to SUPER-INTERVAL syntax.

1 Introduction

Time normalization is the task of translating natural language expressions of time, e.g., *three days ago*, to computer-readable forms, e.g., 2021-11-04. Time normalization is an important component of applications like monitoring patient symptoms (Lin et al., 2015), matching news events across languages (Vossen et al., 2016), and studying date-based literature trends (Fischer and Strötgen, 2015). As part of a larger project studying the causes and effects of food insecurity, we were interested in annotating normalized times in this highly time-sensitive domain.

Several schemes have been proposed for annotating normalized times. TimeML (Pustejovsky et al., 2003; ISO, 2012) primarily focuses on times that can be described as a prefix of YYYY-MM-DDTHH:MM:SS, including relative time expressions like *next Monday*, but excluding time expressions like *the past three summers* that refer to more than one time interval. The semantically compositional annotation for time normalization (SCATE; Bethard and Parker, 2016) scheme breaks time expressions down into individual temporal operators like NEXT or BETWEEN, and can therefore cover a wider variety of time expressions than TimeML. The Time Event Ontology (TEO; Li et al., 2020) draws ideas from both TimeML and SCATE, to provide a simplified annotation scheme specifically targeted at clinical notes.

Because the SCATE scheme covered the widest variety of time expressions, we selected this scheme for annotating normalized times in food security documents. We annotated 17 documents (22K words), producing 2305 time annotations, and achieved an acceptable inter-annotator agreement of 0.808 F1. However, during the process we noticed that to get the correct temporal semantics of expressions like *Nov. 7th to 11th 2021*, we had to add many near-duplicate annotations. Under the SCATE scheme, to get this expression interpreted correctly as [2021-11-07 00:00:00, 2021-11-12 00:00:00), two MONTH-OF-YEAR annotations are needed on *Nov.* and two YEAR annotations are needed on *2021*, as shown in fig. 1 (top). These duplicate annotations are necessary because the BETWEEN needs two intervals, one for 2021-11-07 and one for 2021-11-11, and a YEAR with a chain of SUB-INTERVAL links can represent only a single interval. For example, the top-most YEAR in the figure, based on its chain of SUB-INTERVAL links, is interpreted as 2021-11-11.

Because these types of time expressions were common in our food security documents, we explored a possible change to the SCATE scheme which would replace SUB-INTERVAL links with SUPER-INTERVAL links. In essence, we would reverse the links in the chains, removing the need for the additional annotations, as shown in Figure 1 (bottom). This works because while in the SUB-INTERVAL version each MONTH-OF-YEAR

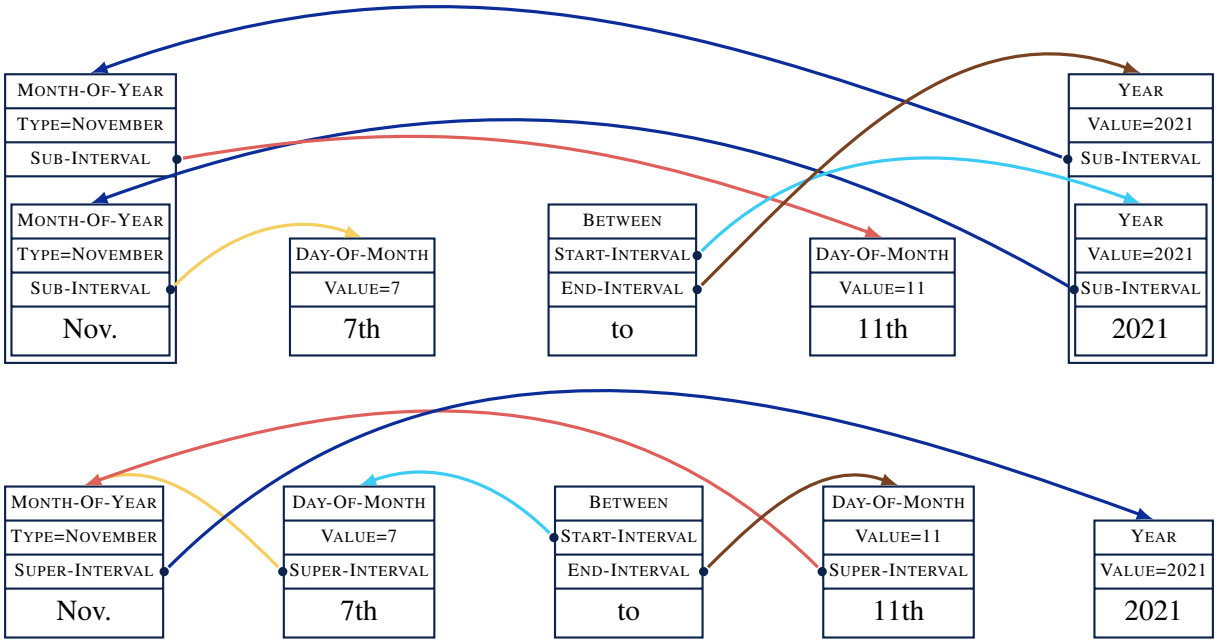


Figure 1: SCATE representation of *Nov. 7th to 11th 2021* using standard SUB-INTERVAL (top) and proposed SUPER-INTERVAL (bottom). Arrows of the same color are performing similar functions.

is linked to a different day and therefore cannot be shared, in the SUPER-INTERVAL version the MONTH-OF-YEAR can be shared because it is linked only to the YEAR, which is also shared.

Our proposed change to SCATE is purely syntactic; if implemented correctly, it should not change the semantics of any complete time expression. But SUB-INTERVAL is extremely common in SCATE-annotated documents, such as the SemEval 2018 Task 6 annotated corpus Laparra et al. (2018), so the conversion from SUB-INTERVAL to SUPER-INTERVAL would have to be automatic, not manual. We thus designed an automatic conversion tool, and a series of validations to ensure that our conversions behaved as expected.

Because SCATE, in addition to defining a set of annotations syntactically, also defines a formal semantics for each annotation, we were able to validate our tool in two different ways: syntactically, ensuring that the XML annotations were valid SCATE annotations and lost no information, and semantically, ensuring that the time intervals inferrable from the SCATE annotations were identical before and after the conversion.

2 Anafora XML to LabelStudio JSON

SCATE was originally implemented in the XML format of the Anafora annotation tool (Chen and Styler, 2013), but that tool has not been actively maintained since 2018. To allow easy visualization of our annotations, we translated the SCATE scheme into a LabelStudio[1] config, and a corresponding LabelStudio JSON format.

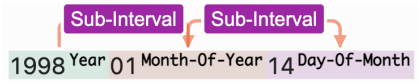
To convert between Anafora XML and LabelStudio JSON, we created an XML-to-JSON script that translated each `<entity>` into a JSON object for its `<type>` and a JSON object for each of its `<properties>`. The structure of these JSON objects follows the LabelStudio requirements. The objects for type and text-based properties have an `id` field, give the name via `from_name`, and give the span and value in the `value` sub-object. The objects for link-based properties have `from_id` and `to_id` fields and give the name via `labels`. Figure 2 shows a sample of the original Anafora XML and the corresponding LabelStudio JSON, as well as a screenshot of LabelStudio visualizing the SCATE annotations.

3 Validating XML-to-JSON

Before implementing the proposed SUB-INTERVAL to SUPER-INTERVAL conversion in the JSON files, we tested our Anafora XML to LabelStudio JSON conversion. We first implemented a JSON-to-XML script designed to be an inverse of the XML-to-JSON script described in section 2, which collected all JSON objects for each `id` and merged them into a single `<entity>`. We then tested that the composition of the two scripts (i.e., XML-to-JSON-to-

[1] https://labelstud.io/

```
<entity>
  <id>51@e@ABC19980114.1830.0611@gold</id>
  <span>29,31</span>
  <type>Month-Of-Year</type>
  <parentsType>Repeating-Interval</parentsType>
  <properties>
    <Type>January</Type>
    <Sub-Interval>50@e@ABC19980114.1830.0611@gold
      </Sub-Interval>
  </properties>
</entity>
```




```
{
  "value": {
    "start": 29, "end": 31,
    "labels": ["Month-Of-Year"]},
  "id": "51@e@ABC19980114.1830.0611@gold",
  "from_name": "type",
},
{
  "value": {
    "start": 29, "end": 31,
    "choices": ["January"]},
  "id": "51@e@ABC19980114.1830.0611@gold",
  "from_name": "Month-Of-Year-type",
},
{
  "from_id": "51@e@ABC19980114.1830.0611@gold",
  "to_id": "50@e@ABC19980114.1830.0611@gold",
  "type": "relation",
  "labels": ["Sub-Interval"]
},
```

Figure 2: Abbreviated example of Anafora XML format (top left), LabelStudio JSON format (right), and LabelStudio visualization (bottom left). See fig. A1 in appendix A for the complete SUB-INTERVAL chain.

```
<entity>
  <id>571@e@ID159_clinic_470@gold</id>
  <span>6491,6495</span>
  <type>After</type>
  <parentsType>Operator</parentsType>
  <properties>
    <Semantics>Interval-Not-Included</Semantics>
    <Interval-Type>Link</Interval-Type>
    <Interval>326@e@ID159_clinic_470@gold</Interval>
  </properties>
</entity>
```

Figure 3: An annotation error. No `<entity>` in the document has id `326@e@ID159_clinic_470@gold`. The phrase *Post* spans the offsets 6491 to 6495 within the phrase *Postoperative*. It is likely that an annotation over the *operative* was lost in the original annotations.

XML) produced Anafora XML that was equivalent to the original Anafora XML. Performing this type of syntactic validation revealed a number of errors.

Annotation errors We found 43 cases in the SemEval 2018 Task 6 annotated corpus where there were errors in the original Anafora XML. For example, fig. 3 shows an `<entity>` whose `<Interval>` appears to link to another entity, but the document contains no entity with id `326@e@ID159_clinic_470@gold`. These were revealed as exceptions in the XML-to-JSON script when it attempted to look up the non-existing entity.

Properties with multiple values Almost all SCATE annotations allow only a single value for each property. However, our XML-to-JSON-to-XML validation revealed that a small number of annotations in the SemEval 2018 Task 6 corpus had properties with multiple values. For example, as shown in fig. 4, *and* in the phrase *Mondays and Tuesdays* was annotated as a UNION with two `<Repeating-Intervals>` arguments, *Mondays* and *Tuesdays*. Our original code assumed one value per property, resulting in missing links in the generated XML. We resolved the issue by ensuring that properties with multiple values resulted in multiple JSON objects instead of just one.

```
<entity>
  <id>130@e@APW19980213.1320@gold</id>
  <span>934,937</span>
  <type>Union</type>
  <parentsType>Operator</parentsType>
  <properties>
    <Repeating-Intervals>111@e@APW19980213.1320@gold
      </Repeating-Intervals>
    <Repeating-Intervals>112@e@APW19980213.1320@gold
      </Repeating-Intervals>
  </properties>
</entity>
```

Figure 4: A `<Repeating-Intervals>` property with multiple values. The phrase *and* spans the offsets 934 to 937 in the phrase *Mondays and Tuesdays*. The first `<Repeating-Intervals>` is an annotation over *Mondays* and the second is an annotation over *Tuesdays*.

Annotations with multiple spans Almost all SCATE annotations are over just a single, continuous span of text. However, our XML-to-JSON-to-XML validation revealed that a small number of annotations in the SemEval 2018 Task 6 corpus had multiple spans. For example, in fig. 5, the event *status, stable* (an event from a clinical note that describes the state of the patient) was annotated to exclude the comma and space characters. Similar problems arise in our food security data, where expressions like *Kiremt 2016 rainy season* needs a discontinuous span for the season *Kiremt . . . rainy season*. We resolved this issue by creating an extra JSON object for each additional span, where the ID is suffixed with `continued` to allow for easy reconstruction of the multi-span annotation.

```
<entity>
  <id>66@e@ID017_clinic_049@gold</id>
  <span>748,754;756,762</span>
  <type>Event</type>
  <parentsType>Other</parentsType>
</entity>
```

Figure 5: An annotation with multiple spans. The phrase *status, stable* spans the offsets 748 to 762, but the comma and space are excluded from the annotation.

4 Validating Sub-to-Super-Interval

Having validated our XML-to-JSON conversion, we implemented our SUB-INTERVAL to SUPER-INTERVAL conversion. Our initial implementation simply changed the `labels` from `Sub-Interval` to `Super-Interval` and swapped the `from_id` and `to_id` fields.

To perform semantic validation of this conversion, we took advantage of the interpretation API provided by Bethard and Parker (2016)[2], which takes annotations as input and produces time intervals as output. For example, the BETWEEN annotation in fig. 1 would be interpreted as the interval [2021-11-07 00:00:00, 2021-11-12 00:00:00). Since the API requires Anafora XML, we used our JSON-to-XML script to convert our SUPER-INTERVAL JSON into SUPER-INTERVAL XML. To check if our syntactic changes to the annotation scheme maintained the intended semantics, we paired up each annotation in the original SUB-INTERVAL XML files with the corresponding annotation in the newly generated SUPER-INTERVAL XML files, and tested that the paired annotations both resulted in the same time intervals. Performing this type of semantic validation revealed a number of errors.

Mid-chain entities The validation showed that the interpretation of entities in the middle of a SUB-INTERVAL chain changed with the SUPER-INTERVAL conversion. For example, the *Nov.* of fig. 1 (top) is intersected with *7th* via its SUB-INTERVAL link, and is therefore interpreted as all November 7th intervals on the timeline. On the other hand, the *Nov.* of fig. 1 (bottom) is intersected with *2021* via its SUPER-INTERVAL link, and is therefore interpreted as the single interval November 2021. This was a desirable consequence of the conversion, since these partial interpretations are necessary to allow re-use of expressions like *Nov.* in *Nov. 7th to 11th*. We therefore changed our semantic validation code to allow mismatches between mid-chain entities.

Entities with chain arguments Our simple swapping of `from_id` and `to_id` turned out to be insufficent for entities that are not part of a SUB-INTERVAL chain, but take one as an argument. For example, the BETWEEN in fig. 1 (top) has chains as its START-INTERVAL and END-INTERVAL arguments. In the SUB-INTERVAL encoding, these chains are represented by YEARs, but in the SUPER-INTERVAL encoding, these chains are represented by DAY-OF-MONTHs. We resolved this issue by finding entities with chain arguments, and replacing the root of the SUB-INTERVAL chain (i.e., the largest time unit) with the root of the SUPER-INTERVAL chain (i.e., the smallest time unit).

5 Discussion

The change from SUB-INTERVAL to SUPER-INTERVAL achieved our primary goal, successfully removing 69 near-duplicate annotations in our food security corpus, without changing the semantics of any time expression. There was only one place where the switch caused a new near-duplicate annotation to be added: *Meher 2016/17*. The *Meher* here refers to two different seasons, May to September 2016 and May to September 2017. With SUB-INTERVALs, both *2016* and *17* could link to the same *Meher* SEASON-OF-YEAR, but with SUPER-INTERVALs, there must be two SEASON-OF-YEARs annotated, each pointing their SUPER-INTERVAL to exactly one of *2016* or *17*. We nonetheless consider the experiment a success: 69 near-duplicates removed, at the cost of just 1 near-duplicate added.

In terms of validation strategies, we found it helpful to be working with an annotation scheme that had both a syntactic specification of how annotations should be applied to words in the text, and a formal semantic interpretation that converted the annotations to explicit intervals on the timeline. This allowed us to change the syntax of the annotation scheme, while making sure that we did not unintentionally change the semantics of the annotations. Having both a syntactic specification and a formal semantic interpretation is uncommon in annotation schemes. At one end of the spectrum are purely syntactic annotation schemes, like Universal Dependencies (UD; Nivre et al., 2020), with no formal semantic interpretation at all. At the other end of the spectrum are purely semantic annota-

[2] https://github.com/clulab/timenorm/

tion schemes, like abstract meaning representation (AMR; Banarescu et al., 2013) where only the formal logic-like interpretation is annotated, with no explicit links to individual words of the original text. Our experience was that having both a syntactic and semantic specification made it easier to apply and validate improvements to the annotation scheme.

All code produced during this work can be found at https://github.com/clulab/timenorm/.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Steven Bethard and Jonathan Parker. 2016. A semantically compositional annotation scheme for time normalization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3779–3786, Portorož, Slovenia. European Language Resources Association (ELRA).

Wei-Te Chen and Will Styler. 2013. Anafora: A web-based general purpose annotation tool. In *Proceedings of the 2013 NAACL HLT Demonstration Session*, pages 14–19, Atlanta, Georgia. Association for Computational Linguistics.

Frank Fischer and Jannik Strötgen. 2015. When Does (German) Literature Take Place? On the Analysis of Temporal Expressions in Large Corpora. In *Proceedings of DH 2015: Annual Conference of the Alliance of Digital Humanities Organizations*, volume 6, Sydney, Australia.

ISO. 2012. Language resource management – semantic annotation framework (semaf) – part 1: Time and events (semaf-time, iso-timeml). Technical report, ISO. 24617-1:2012.

Egoitz Laparra, Dongfang Xu, Ahmed Elsayed, Steven Bethard, and Martha Palmer. 2018. SemEval 2018 task 6: Parsing time normalizations. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 88–96, New Orleans, Louisiana. Association for Computational Linguistics.

Fang Li, Jingcheng Du, Yongqun He, Hsing-Yi Song, Mohcine Madkour, Guozheng Rao, Yang Xiang, Yi Luo, Henry W Chen, Sijia Liu, Liwei Wang, Hongfang Liu, Hua Xu, and Cui Tao. 2020. Time event ontology (TEO): to support semantic representation and reasoning of complex temporal relations of clinical events. *Journal of the American Medical Informatics Association*, 27(7):1046–1056.

Chen Lin, Elizabeth W. Karlson, Dmitriy Dligach, Monica P. Ramirez, Timothy A. Miller, Huan Mo, Natalie S. Braggs, Andrew Cagan, Vivian S. Gainer, Joshua C. Denny, and Guergana K. Savova. 2015. Automatic identification of methotrexate-induced liver toxicity in patients with rheumatoid arthritis from the electronic medical record. *Journal of the American Medical Informatics Association*, 22(e1):e151–e161.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

James Pustejovsky, José M. Castaño, Robert Ingria, Roser Saurí, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. 2003. Timeml: Robust specification of event and temporal expressions in text. In *New Directions in Question Answering, Papers from 2003 AAAI Spring Symposium, Stanford University, Stanford, CA, USA*, pages 28–34. AAAI Press.

Piek Vossen, Rodrigo Agerri, Itziar Aldabe, Agata Cybulska, Marieke van Erp, Antske Fokkens, Egoitz Laparra, Anne-Lyse Minard, Alessio Palmero Aprosio, German Rigau, Marco Rospocher, and Roxane Segers. 2016. NewsReader: Using knowledge resources in a cross-lingual reading machine to generate more knowledge from massive streams of news. *Special Issue Knowledge-Based Systems, Elsevier*.

A Appendix

Figure A1 shows an extended example of what SUB-INTERVAL chains look like in both Anafora XML and LabelStudio JSON.

```
<entity>
  <id>50@e@ABC19980114.1830.0611@gold</id>
  <span>31,33</span>
  <type>Day-Of-Month</type>
  <parentsType>Repeating-Interval</parentsType>
  <properties>
  <Value>14</Value>
  <Sub-Interval></Sub-Interval>
  <Number></Number>
  <Modifier></Modifier>
  </properties>
</entity>
<entity>
  <id>51@e@ABC19980114.1830.0611@gold</id>
  <span>29,31</span>
  <type>Month-Of-Year</type>
  <parentsType>Repeating-Interval</parentsType>
  <properties>
  <Type>January</Type>
  <Sub-Interval>50@e@ABC19980114.1830.0611@gold
    </Sub-Interval>
  <Number></Number>
  <Modifier></Modifier>
  </properties>
</entity>
<entity>
  <id>52@e@ABC19980114.1830.0611@gold</id>
  <span>25,29</span>
  <type>Year</type>
  <parentsType>Interval</parentsType>
  <properties>
  <Value>1998</Value>
  <Sub-Interval>51@e@ABC19980114.1830.0611@gold
    </Sub-Interval>
  <Modifier></Modifier>
  </properties>
</entity>
```

```
{
  "value": {
    "start": 31,
    "end": 33,
    "labels": ["Day-Of-Month"]
  },
  "id": "50@e@ABC19980114.1830.0611@gold",
  "from_name": "type",
  "to_name": "text",
  "type": "labels"
},
{
  "value": {
    "start": 31,
    "end": 33,
    "text": ["14"]
  },
  "id": "50@e@ABC19980114.1830.0611@gold",
  "from_name": "Day-Of-Month-value",
  "to_name": "text",
  "type": "textarea"
},
{
  "value": {
    "start": 29,
    "end": 31,
    "labels": ["Month-Of-Year"]
  },
  "id": "51@e@ABC19980114.1830.0611@gold",
  "from_name": "type",
  "to_name": "text",
  "type": "labels"
},
{
  "value": {
    "start": 29,
    "end": 31,
    "choices": ["January"]
  },
  "id": "51@e@ABC19980114.1830.0611@gold",
  "from_name": "Month-Of-Year-type",
  "to_name": "text",
  "type": "choices"
},
{
  "from_id": "51@e@ABC19980114.1830.0611@gold",
  "to_id": "50@e@ABC19980114.1830.0611@gold",
  "type": "relation",
  "labels": ["Sub-Interval"]
},
{
  "value": {
    "start": 25,
    "end": 29,
    "labels": ["Year"]
  },
  "id": "52@e@ABC19980114.1830.0611@gold",
  "from_name": "type",
  "to_name": "text",
  "type": "labels"
},
{
  "value": {
    "start": 25,
    "end": 29,
    "text": ["1998"]
  },
  "id": "52@e@ABC19980114.1830.0611@gold",
  "from_name": "Year-value",
  "to_name": "text",
  "type": "textarea"
},
{
  "from_id": "52@e@ABC19980114.1830.0611@gold",
  "to_id": "51@e@ABC19980114.1830.0611@gold",
  "type": "relation",
  "labels": ["Sub-Interval"]
}
```

Figure A1: Example Anafora XML format (left) and LabelStudio JSON format (right).