SUTIME: Evaluation in TempEval-3

Angel X. Chang
Stanford University
angelx@cs.stanford.edu

Christopher D. Manning
Stanford University
manning@cs.stanford.edu

Abstract

We analyze the performance of SUTIME, a temporal tagger for recognizing and normalizing temporal expressions, on TempEval-3 Task A for English. SUTIME is available as part of the Stanford CoreNLP pipeline and can be used to annotate documents with temporal information. Testing on the TempEval-3 evaluation corpus showed that this system is competitive with state-of-the-art techniques.

1 Introduction

The importance of modeling temporal information is increasingly apparent in natural language applications, such as information extraction and question answering. Extracting temporal information requires the ability to recognize temporal expressions, and to convert them from text to a normalized form that is easy to process. Temporal tagging systems are designed to address this problem. In this paper, we evaluate the performance of the SUTIME (Chang and Manning, 2012) rule-based temporal tagging system.

We evaluate the performance of SUTIME on extracting temporal information in TempEval-3 (UzZaman et al., 2013), which requires systems to automatically annotate documents with temporal information using TimeML (Pustejovsky et al., 2003). The TempEval-3 training data contains gold human annotated data from TimeBank, AQUAINT, and a new dataset of silver data automatically annotated using a combination of TipSem (Llorens et al., 2010) and TRIOS (UzZaman and Allen, 2010), two of the best performing systems from TempEval-2 (Verhagen et al., 2010).

2 System Description

We use the Stanford CoreNLP[1] pipeline with SUTIME to identify and normalize TIMEX3[2] expressions. SUTIME is incorporated into Stanford CoreNLP as part of the Named Entity Recognition annotator. For TempEval-3, we use the standard set of rules provided with SUTIME. Since SUTIME can also recognize temporal expressions whose values are not specified by TIMEX3, we ran SUTIME in a TIMEX3 compatible mode.[3]

2.1 SUTime

SUTIME is a rule-based temporal tagger built on regular expression patterns over tokens. Temporal expressions are bounded in their complexity, so many of them can be captured using finite automata. As shown by systems such as FASTUS (Hobbs et al., 1997), a cascade of finite automata can be very effective at extracting information from text. With SUTIME, we follow a similar staged strategy of (i) building up patterns over individual words to find numerical expressions; then (ii) using patterns over words and numerical expressions to find simple temporal expressions; and finally (iii) forming composite patterns over the discovered temporal expressions.

SUTIME recognizes **Time**, **Duration**, **Interval**, and **Set** according to the TIMEX3 specification. In

[1] nlp.stanford.edu/software/corenlp.shtml
[2] www.timeml.org
[3] sutime.restrictToTimex3 = true

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 78–82, Atlanta, Georgia, June 14-15, 2013.

addition, it recognizes nested time expressions and duration ranges. To achieve this it uses a temporal pattern language defined over tokens (a regular expression language for expressing how tokenized text should be mapped to temporal objects). SUTIME is built on top of TOKENSREGEX,[4] a generic framework included in Stanford CoreNLP for defininig patterns over text and mapping to semantic objects. With TOKENSREGEX we have access to any annotations provided by the Stanford CoreNLP system, such as the part-of-speech tag or the lemma. The full specification of the pattern language is available at `nlp.stanford.edu/software/sutime.shtml`.

To recognize temporal expressions, SUTIME applies three types of rules, in the following order: 1) text regex rules: mappings from simple regular expressions over characters or tokens to temporal representations; 2) compositional rules: mappings from regular expressions over chunks (both tokens and temporal objects) to temporal representations and 3) filtering rules: in which ambiguous expressions that are likely to not be temporal expressions are removed from the list of candidates (such as *fall* and *spring* by themselves). The compositional rules are applied repeatedly until the final list of time expressions stablizes.

After all the temporal expressions have been recognized, each temporal expression is associated with a temporal object. Each temporal object is resolved with respect to the reference date using heuristic rules. In this step, relative times are converted to an absolute time, and composite time objects are simplified as much as possible. The final resolution of relative temporal expressions is currently limited due to the usage of simple hard-coded rules (e.g. relative to document date with local context informing before and after heuristics). Finally, SUTIME will take the internal time representation and produce a TIMEX3 annotation for each temporal expression. SUTIME currently only handles English. It can however, be extended to other languages by creating sets of rules for additional languages.

3 Evaluation

We evaluated SUTIME's performance on the TempEval-3 Task A for English. Task A consists of determining the extent of time expressions as defined by the TimeML TIMEX3 tag, as well as providing normalized attributes for *type* and *value*. Extracted temporal expressions from the system and the gold are matched, and precision, recall, and F_1 are computed. For the evaluation of extents, there are two metrics: a relaxed match score for identifying a matching temporal expression, and a strict match that requires the text to be matched exactly. For example, identifying *the twentieth century* when the gold is *twentieth centry* will give a relaxed match but not a strict match. For the type and value attributes, an accuracy and a measure of the F_1 with respect to the relaxed match is given.

We compare SUTIME's performance with several other top systems on the English TempEval-3 Task A. We also include TIPSem which was used to create the silver data for TempEval-3 as a baseline. Of the systems that prepared multiple runs, we selected the best performing run to report. Table 1 gives the results for these systems on the TempEval-3 evaluation set. Interestingly, NavyTime which uses SUTIME for Task A, actually did better than SUTIME in the value normalization and is effectively the 2nd best system in Task A. The performance of NavyTime is otherwise identical to SUTIME. In NavyTime the normalization was tuned to the TimeBank annotation whereas the SUTIME submission was untuned. SUTIME has the highest recall in discovering temporal expressions. It also has the highest overall relaxed F_1, slightly higher than HeidelTime (Strötgen and Gertz, 2010) (cleartk had the highest strict F_1 of 82.71). Not surprisingly, the system used to generate the silver data, TIPSem, had the highest precision when extracting temporal expressions. For normalization, HeidelTime had the overall best performance on value and type. Both SUTIME and HeidelTime are rule-based, indicating the effectiveness of using rules for this domain. Another top performing system, ManTime used conditional random fields, a machine learning approach, for identifying temporal expressions and rules for normalization.

[4] `nlp.stanford.edu/software/tokensregex.shtml`

	Identification						*Normalization*			
	Relaxed			*Strict*			*Value*		*Type*	
System	F_1	P	R	F_1	P	R	F_1	Accuracy	F_1	Accuracy
SUTime	**90.32**	89.36	**91.30**	79.57	78.72	**80.43**	67.38	74.60	80.29	88.90
NavyTime	**90.32**	89.36	**91.30**	79.57	78.72	**80.43**	70.97	78.58	80.29	88.90
HeidelTime	90.30	93.08	87.68	81.34	83.85	78.99	**77.61**	**85.95**	**82.09**	**90.91**
ManTime	89.66	95.12	84.78	74.33	78.86	70.29	68.97	76.92	77.39	86.31
TIPSem	84.90	**97.20**	75.36	81.63	**93.46**	72.46	65.31	76.93	75.92	89.42

Table 1: TempEval-3; English Platinum Test set.

4 Error Analysis

Given the small size of the platinum data set, we were able to perform thorough error analysis of the errors made by SUTIME on the data set.

Table 2 shows the number of temporal expressions marked by the evaluation script as being incorrect. The errors can be grouped into three broad categories: i) those proposed by the system but not in the gold (relaxed precision errors), ii) those in the gold but not identified by the system (relaxed recall errors), and iii) temporal expressions with the wrong value (and sometimes type) normalization.

Of the 14 precision errors, many of the temporal expressions suggested by the system are reasonable. For instance, *current* is identified by the system. A few of the errors are not actual temporal expressions. For example, in the phrase *British Summer Time*, *Summer* was identified as a temporal expression which is not correct.

Given SUTime's high recall, only a few temporal expressions in the gold are not found by the system. In most cases, the temporal expressions missed by SUTIME do not have a well defined value associated with them (e.g. "digital age", "each season").

Performance using the strict match metric is not as good as some other systems. SUTIME was derived from GUTime (Mani, 2004) and focuses on matching longer time expressions as per earlier guidelines. Thus it is less conformant to the more current TimeML guidelines of having minimal blocks. For instance, SUTIME treats 2009-2010 as a range, whereas the gold standard treats it as two separate dates. This results in an incorrect value normalization and a recall error.

We now examine the cases where the SUTIME normalization differed from the gold. Table 3 shows a further breakdown of these errors.

Error type	Count
System not in gold (precision)	14
Gold not in system (recall)	12
Wrong value	32

Table 2: Summary of errors made by SUTIME on the platinum data set

Error type	Count
Value incorrectly resolved wrt to DCT	7
Value should not be resolved wrt to DCT	5
DURATION resolved to DATE	6
DATE misidentified as DURATION	3
Wrong granularity	4
Wrong normalization for set	2
Different normalization	3
Other	2

Table 3: Break down of value errors made by SUTime on the platinum data set

One weakness of SUTIME is that temporal expressions are always resolved with respect to the document creation time (DCT). While this heuristic works fairly well in most cases, and SUTime can achieve reasonable performance, there are obvious limitations with this approach. For instance, sometimes it is more appropriate to resolve the temporal expression with respect to nearby dates or events in the text. As an example, in the test document CNN_20130322_1003 there is the sentence *Call me Sunday night at 8 PM at the resort* that is part of an email of an unknown date. In this case, SUTIME still attempts to resolve the temporal expression *Sunday night at 8 PM* using the document creation time which is incorrect.

There can be inherent ambiguity as to which time point a time expression refers to. For instance, given a reference date of **2011-09-19**, a Monday, it is un-

clear whether *Friday* refers to **2011-09-16** or **2011-09-23**. SUTIME will normally resolve to the closest date/time with respect to the reference date. SUTIME also has some rules that will use the verb tense of the surrounding words to attempt to resolve the ambiguity. For instance, if a verb close to the temporal expression has a POS tag of VBD (past tense verb) then the expression will be resolved so that it occurs before the document date.

Most of the type errors are due to confusions between DATE and DURATION. Often SUTIME will attempt to resolve a DURATION as a DATE. For instance, given the phrase "the following decade", SUTIME will attempt to resolve that as a DATE with value **202X** (using a DCT of 2013-03-22). While this can be desirable in some cases, this is not what the gold annotation contains: type of DURATION and value of **P1DE**. In some other cases, SUTIME misidentifies DURATION as a DATE. For instance, it lacks rules to parse the *3:07:35* in *finishing in 3:07:35* as a duration.

Another problem faced by SUTIME is in figuring out the correct granularity to use. Given a document date of **2013-03-22**, it will identify *two years ago* as being **2011-03-22**. However, since these expressions indicate a less precise date, the gold annotation is a simple **2011**.

SUTIME also provided the wrong normalization for SET in several cases. For the expression *every morning*, SUTIME reported a value of **TMO** when the gold annotation was **XXXX-XX-XXTMO**. In other cases, SUTIME offered an alternative normalization, for instance, a value of **19XX** for *the 20th century* instead of just **19**. And **PTXM** instead of **PXM** for *minutes*. In this case, the **PTXM** is more correct as the **T** is required by ISO-8601 to differentiate between **M** for month, and **M** for minutes. The remaining errors are due to lacking rules such as SUTIME's inability to handle time zones in certain cases.

5 Discussion

As a rule-based system, SUTIME is limited by the coverage of its rule set for the different types of temporal expressions it can recognize. Many of the errors in SUTIME can be resolved by adding more rules to the system.

One key to improving the normalization of the value is to have better resolution of ambiguous temporal expressions. Identifying when temporal expressions should not be resolved using the document creation time, and how the temporal expression relates to other temporal expressions or events within the document is also critical. This suggests that normalization can benefit from being able to perform TempEval-3 Task C well.

Another approach to improving the system would be to provide different modes of use: a mode for end users that would like complex temporal expressions to be identified, or a mode for more basic temporal expressions that can be used as input for other temporal systems. Allowing for nested TIMEXes would also benefit the system's performance. For example, *2009-2010* should be a range, with a nested timex for *2009* and *2010*.

Another interesting direction to explore would be to evaluate the performance of SUTIME on domains other than current news. Since SUTIME also supports temporal expressions such as holidays and more distant dates such as *400 B.C.*, it would be interesting to see how well SUTIME can extract these different types of temporal expressions.

6 Conclusion

We have evaluated SUTIME by participating in TempEval-3 Task A and have shown that it is a competitive system for extracting time expressions. By providing it as part of the Stanford CoreNLP pipeline, we hope that it can be easily used as a basic component for building temporally aware systems.

Acknowledgements

We would like to acknowledge Justin Heermann, Jonathan Potter, Garrett Schlesinger and John Bauer for helping to implement parts of the system for TempEval-3.

References

Angel X. Chang and Christopher D. Manning. 2012. SUTIME: A library for recognizing and normalizing time expressions. In *8th International Conference on Language Resources and Evaluation (LREC 2012)*.

Jerry R. Hobbs, Douglas E. Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel, and Mabry

Tyson. 1997. FASTUS: A cascaded finite-state transducer for extracting information from natural-language text. *Finite State Devices for Natural Language Processing*, pages 383–406.

Hector Llorens, Estela Saquete, and Borja Navarro. 2010. TIPSem (English and Spanish): Evaluating CRFs and semantic roles in TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291. Association for Computational Linguistics.

Inderjeet Mani. 2004. Recent developments in temporal information extraction. In *Proceedings of RANLP03*, pages 45–60.

James Pustejovsky, Jos Castao, Robert Ingria, Roser Saur, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. TimeML: Robust specification of event and temporal expressions in text. In *in Fifth International Workshop on Computational Semantics (IWCS-5*.

Jannik Strötgen and Michael Gertz. 2010. HeidelTime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324.

Naushad UzZaman and James F Allen. 2010. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, Leon Derczynski, Marc Verhagen, James Allen, and James Pustejovsky. 2013. SemEval-2013 Task 1: TempEval-3: Evaluating time expressions, events, and temporal relations. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, SemEval '13.

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