Temporal Information Extraction using Regular Expressions

Anton Fagerberg
D10, Lund Institute of Technology, Sweden
anton@antonfagerberg.com
ada10afa@student.lu.se
2014-01-13

Abstract

This is a description & a evaluation of a proof-of-concept temporal tagger – a system designed to extract temporal information using regular expressions and apply normalisation functions to them. The system was developed with TempEval-2 task A (Pustejovsky et al., 2009), a part of the SemEval-2010 task #13, as its basis.

1 Introduction

1.1 Temporal information extraction

Temporal information extraction involves two main tasks. First of all, the temporal expressions has to be recognised from some kind machine-readable document and extracted from it.

... on **March 29, -86** he visited the ...
... who concluded **last month** that ...
Two weeks after Arafat's death ...

Figure 1: Excerpts from news articles with temporal expressions highlighted.

When a temporal expression has been recognised and extracted, it should be categorised and normalised to a canonical form. Unless the normalisation isn't just a question of formatting, the process may include some kind of calculation to determine the appropriate value of the extracted expression.

Expression	Type	Normalised
March 29, -86	Date	1986-03-29
last month	Date	1998-12
Two weeks	Duration	P2W

Figure 2: Categorisation and normalisation of extracted data.

1.2 TempEval-2

This project builds heavily upon task A of TempEval-2 (Pustejovsky et al., 2009), a part of the SemEval International Workshop on Semantic Evaluations where TempEval-2 was task #13 of SemEval-2010. Task A of TempEval-2 is defined as:

"Determine the extent of the time expressions in a text as defined by the TimeML timex3 tag. In addition, determine value of the features type and val. The possible values of type are time, date, duration, and set; the value of val is a normalized value as defined by the timex2 and timex3 standards."

DATE	**DURATION**
Friday, October 1, 1999	2 months
the second of December	48 hours
yesterday	three weeks
TIME	**SET**
ten minutes to three	twice a week
at five to eight	every 2 days
at twenty after twelve	once a year

Figure 3: Types with examples as defined by TempEval-2.

2 Method

2.1 Extraction

To recognise temporal expressions, a hierarchy of regular expressions was used. A sorted hierarchy has to be defined where the more specific rules takes precedence over the more general ones in order to avoid faulty matches, see Figure 4. The construction of such a hierarchy is initially easy but it can get complicated to maintain and extend when the regular expressions start to interfere with each other in ways that are hard to predict.

Expression	**Type**	**Value**
two weeks ago	Date	2013-W45
two weeks	Duration	P2W

Figure 4: Two similar regular expressions are needed but the one on top needs higher precedence since it is more specific.

2.2 Normalisation

When an expression has been extracted, it has to be categorised and normalised. In the system developed, the hierarchy of regular expressions was extended to be a hiearchy of three-tuples consisting of a regular expression, the category (type) and a function which transformed the matched expression to the normalised form using TIMEX3 tags (TimeML Working Group, 2009).

The normalisation functions varied a lot in complexity. In some cases the normalisation function only involved a simple rewrite such as transforming *"21/5 -98"* to *"21-05-1998"* – but other functions required additional calculations and context about the input source to produce a valid result.

When evaluating an expression such as *"on thursday"*, an additional context is required to determine the actual date of the thursday referenced. First of all, we need to know the publication date of the text to have a reference point. In addition, we might be given additional clues based on what medium the actual source is.

Since the TempEval-2 English training and test corpus was composed of news articles, we can hypothesise that *"on thursday"* refers to the first thursday in the past in relation to the publication date. It is a generalisation but it does hold true in a vast majority of the cases since the news, most of the time, reports what has happened and not what is going to happen. However, if the input data was in an other format, such as SMS text messages, using the following thursday in respect to the receiving date may be more appropriate.

2.3 Iteration

With the regular expression, type and normalisation function three-tuples in place, the input text has to be broken down to workable pieces. A form of sliding window was used for this.

Suppose we have a news article where the following text is present somewhere within it *"suggested June last year for continuing negotiations"*.

The text would then iterated by taking the first five words, trying to match everything in the window with any of the regular expressions in the hierarchy. If every matching fails, remove the last word from the current window and try again until the window is empty. If no matches was found, the window is moved one step further, encapsulating the next five words.

Iteration 1
suggested June last year for
suggested June last year
suggested June last
suggested June
suggested
Iteration 2
June last year for continuing
June last year for
June last year
~~June last~~
~~**June**~~

Figure 5: Two matches are found in the second iteration – only the first one is used. The following two iterations will be ignored as well to avoid matching *last year* in iteration 3 and *year* in iteration 4.

When a valid match is found somewhere inside a window iteration it is processed and the rest of the iteration is discarded. If the expression we have matched has n words, the following $n - 1$ window movements are also discarded to avoid the matching of smaller expressions inside an already matched larger expression.

3 Result

3.1 Comparison

The system was during development evaluated against the TempEval-2 English training corpus. When the deadline for the project was reached, it was evaluated against the English test corpus for a final review. As a comparison, results from the best performing contestants along with data provided by SUTime (Chang and Manning, 2012) has been added to the table in Figure 6. A comprehensive review of all actual contestants in TempEval-2 can be found in (Verhagen et al., 2010).

Name	P	R	F_1	A_t	A_v
Fagerberg	**0.95**	**0.61**	**0.75**	**0.96**	**0.88**
GUTime	0.89	0.79	0.84	0.88	0.96
SUTime	0.92	0.85	0.85	0.85	0.90
TRIPS/TRIOS	0.82	0.86	0.82	0.91	0.86
HeidelTime1	0.57	0.89	0.70	0.95	0.68
HeidelTime2	0.96	0.82	0.94	0.76	0.96
HeidelTime*	0.85	0.92	0.77	0.96	0.85

Figure 6: Results from top contestants in TempEval-2 and additional data provided by SUTime (Chang and Manning, 2012), English evaluation set. The results from this paper are named *Fagerberg*. A_t = attribute type, A_v = attribute value. Note that A_v and A_t is simply $correct/answers$ – no penalty is given for no answer.

precision = tp / (tp + fp)
recall = tp / (tp + fn)
accuracy = (tp + tn) / (tp + tn + fp + fn)
f1-measure = 2 * (prec * rec) / (prec + rec)

Figure 7: How the scores are calculated. True positives (tp), false positives (fp), true negatives (tn), false negatives (fn), recall (rec), precision (prec).

3.2 Alaysis

As the table indicates; the precision is very good while the recall is lacking. In order to increase the recall, additional three-tuples has to be added to the hierarchy without penalising the precision. We can also see that the attribute value A_v and attribute type A_t are good but the result is somewhat misleading since it only counts actual guesses and does not penalise ignored expressions.

The lacking recall is affected by the fact that, due to time constraints, neither the type *Time* nor *Set* had any matching rules at all on evaluation. Providing some basic matching rules for these should improve the recall score additionally without having to worry about interfering with existing rules.

4 Conclusions

4.1 Review

As the results indicates, using three-tuples of regular expressions, type and normalisation functions in a hierarchy can be very effective. The best performant of this task in 2010, Heideltime (Strötgen and Gertz, 2010), uses an approach very similar to this, as does the more recent SUTime (Chang and Manning, 2012) which further proves its effectiveness. However, TRIPS/TRIOS which is a probabilistic system, was also very effective and shows that other approaches are worth examining as well.

The system developed was, because of time constraints, more of a proof-of-concept rather than a functional system and does therefor lack some features and most notably doesn't support anything of type *Time* or *Set*. Given additional time, it should be possible to improve the recall a lot by expanding the three-tuple hierarchy with additional rules – with the help of the TempEval 2 training set.

4.2 Limitations & Improvements

There exist some very complex expressions, which requires a lot of specific rules to cover all of them – such as variations of *"a rate of 358,000 a month for the last four months"* from training corpus *NYT19980206.0460*. Covering every such nested combination of cases may not be feasible in this approach.

A limitation with only using regular expression arises from a previously noted case when an expression such as *"on thursday"* should be evaluated. The possibility to analyse the sentence for additional clues, such as looking at the verb tense, and not make a qualified guess about which thursday is referenced, would improve the precision additionally – especially when the input source is not as predictable as news articles.

Additional sentence analysis would also be helpful when looking at ambiguous words such as *March* which could be the name of a month or the walking of military troops. Some precautions have been taken such as only using *March* as a month when it has a capital M. This does improve the precision but is far from a complete solution since it does not apply to every language as well as the fact that sentences always start with a capital letter.

Another possible improvement would be expanding the window size to account for more than five words. Evaluation has shown that such an expansion will not yield any significant improvements, at least with the current rule hierarchy and input sources. However, when adding more complex rules or working on other input souces, such an expansion might be necessary.

It should finally be noted that most of the regular expressions specifically targets the English language and will not work on any other language. If the system should be extended to additional languages, it would be hard to adapt the existing rules with respect to the existing hierarchy.

5 Acknowledgements

Big thanks to my mentor Pierre Nugues who coordinated the course at LTH in which this project took place and who also provided invaluable feedback, reading materials and introduced me to TempEval-2. Thanks to Håkan Jonsson from Sony Mobile who also helped mentoring me during the project. I would also like to gratefully acknowledge Lund University, faculty of engineering, LTH, for providing the course EDAN60, Language Technology: Project, which let me explore this topic.

References

Angel X. Chang, and Christopher D. Manning 2012. *SUTIME: A Library for Recognizing and Normalizing Time Expressions.*

James Pustejovsky, Marc Verhagen, Xue Nianwen, Robert Gaizauskas, Mark Hepple, Frank Schilder, Graham Katz, Roser Saur , Estela Saquete, Tommaso Caselli, Nicoletta Calzolari, Kiyong Lee, and Seohyun Im. 2009. *TempEval2, SemEval Task Proposal, Evaluating Events, Time Expressions and Temporal Relations.*

Jannik Strötgen and Michael Gertz. 2009. *HeidelTime: High Qualitiy Rule-based Extraction and Normalization of Temporal Expressions.*

TimeML Working Group. 2009. *Guidelines for Temporal Expression Annotation for English for TempEval 2010.*

Marc Verhagen, Roser Saur, Tommaso Caselli and James Pustejovsky 2010. *SemEval-2010 Task 13: TempEval-2.*