|  |
| --- |
| TERNIP: Temporal Expression Recognition and Normalisation in Python |
| INTERIM REPORT  COM6920 Thesis Preparation |
| Christopher Northwood  *Supervisor: Mark Hepple* |

Table Of Contents

[Table Of Contents 3](#_Toc260236485)

[1 Introduction 4](#_Toc260236486)

[2 Background 5](#_Toc260236487)

[2.1 Temporal Expressions 5](#_Toc260236488)

[2.2 Annotation Standards 6](#_Toc260236489)

[2.3 Temporal Expression Taggers 6](#_Toc260236490)

[2.4 Evaluating Tagger Performance 8](#_Toc260236491)

[2.5 Rule Discovery 8](#_Toc260236492)

[3 Project Aims 9](#_Toc260236493)

[4 Work Plan 10](#_Toc260236494)

[4.1 Overview 10](#_Toc260236495)

[4.2 Python Tagger 10](#_Toc260236496)

[4.3 Automatic Rule Discovery 10](#_Toc260236497)

[4.4 Evaluation 10](#_Toc260236498)

[4.5 Timescale 11](#_Toc260236499)

[5 Bibliography 12](#_Toc260236500)

# Introduction

In this report, the outline of a system for temporal expression recognition and normalisation is presented, called TERNIP – Temporal Expression Recognition and Normalisation in Python.

Temporal expressions are words and phrases which refer to some point in time (Ahn, Rantwijk, & de Rijke, 2007), and the distinct, but related, tasks of recognition and normalisation refer to the identification and resolution of these expressions to some standard format in time. A more detailed overview of temporal expressions and the development of the fields of recognition and normalisation is given in section 2.1.

A number of systems and approaches have been taken to the tasks of recognition and normalisation, a survey of which appears in section 2.3. Additionally, a number of standards for annotation have been defined which is covered in section 2.2, and the Time Expression Recognition & Normalization evaluation (MITRE, 2004) provides a corpus and some comparative results, which is outlined in section 2.4.

TERNIP aims to build on work already done, specifically that of the GUTime tagger (Verhagen, et al., 2005), by building a system for the recognition and normalisation of temporal expressions in Python (van Rossum, 1995). The more detailed aims of the project are outlined in section 3.

Finally, section 4 proposes a series of tasks which will be undertaken by the project to meet the stated aims, along with a plan for the execution of these tasks.

# Background

## Temporal Expressions

Temporal expressions, or “timexes”, are “phrases or words that refer to times, where times may be points or durations, or sets of points or durations” (Ahn, Rantwijk, & de Rijke, 2007) , and the identification and interpretation of these timexes is an active topic of research.

Most systems deal with two distinct, but related, tasks for the identification of timexes. The first is that of recognition, which simply identifies which phrases in some text are temporal, that is, refer to some point in time. The second task is that of normalisation, which takes the identified expressions, and attempts to resolve it into some standard format (e.g., ISO 8601) to anchor the expression at a particular point in time (Ahn, Adafre, & de Rijke, Recognizing and Interpreting Temporal Expressions in Open Domain Texts, 2005).

Interest in recognition of temporal expressions grew out of the field of information extraction. The Message Understanding Conferences of the 1990s dealt with the tasks of named entity recognition, and early timex recognition systems simply dealt timex recognition as a part of named entity recognition (Krupka & Hausman, 1998). Temporal expression recognition is clearly an important task for information extraction; however identification of temporal expressions by itself is of limited usefulness.

Normalisation is important to allow for further processing, such as construction of event chronologies, or in question answering systems, and is an important part of natural language understanding. In the phrase “Do you want to go to the pub at 7?”, a human may normalise the expression “7” to a particular point in time based on context of the current date, and the background knowledge that visits to public houses are more likely in the evening.

(Mani & Wilson, 2000) was a prominent, early system which used a rule-based system for normalisation based on establishing tense. Following this, the Time Expression Recognition and Normalization (TERN) evaluation as part of the 2004 Automated Content Extraction (ACE) programme (MITRE, 2004) was the first competition that dealt specifically with recognition and normalisation as a distinct task from named entity recognition.

Following this early work and the TERN competition, interest in temporal expressions has grown, with multiple systems and approaches to recognition and normalisation taken. These systems and approaches are discussed further in section 2.3.

Simple normalisation of temporal expressions is not enough to capture the full range of temporal information available in a body of text, as much temporal information is implicit (Verhagen, 2004). For example, in the phrase “A goal was scored shortly after kick-off”, there is no explicit temporal information there, but there is some implicit information that could be obtained. In this case, the events of the goal being scored and kick-off are identified, and there is a temporal ordering between them, as well as implicit temporal information in these events themselves.

Much recent research has been done on identifying and annotating temporal relations, which build on top of temporal expression recognition and normalisation, however effective temporal recognition and normalisation is still required for this work to be effective.

## Annotation Standards

A number of standards for annotation of temporal expressions have emerged over time. The first annotation formats were typically based on SGML and XML and were simply in a format decided by the tagger. Over time, a standardisation effort for annotation emerged, culminating in TimeML.

TimeML is an XML-based annotation language, complete with a set of guidelines for timex annotation based on the earlier .... standards.

<<< Overview of TimeML, with reference to earlier standards >>>

As interest in temporal expressions has grown to include event identification and temporal relations, the TimeML standard also includes tags and annotation guidelines for more than just timexes, such as events, <<< ETC >>>

<<< Talk about non-TIMEX tags >>>

<<< Annotation guidelines >>>

## Temporal Expression Taggers

Temporal expression taggers are tools which annotate the timexes in some input text. Early taggers focussed just on identification with normalisation becoming incorporated into later systems.

<<< Investigations to be done on hand-tagging: Setzer and Rob G >>>

The tasks of automated recognition and normalisation is often rolled into the same tool, although (Ahn, Adafre, & de Rijke, 2005) argue that separation of these components is beneficial.

<<< typically marked up with POS tags? >>>

<<< Recognition trivial, grew out of entity recognition >>>

The earliest automated temporal expression annotation systems treated temporal expression recognition as a task along with entity recognition , and used simple hand-written rules . In both systems, grammars were provided for the named entity recognisers and the time expressions simply recognised. No normalisation was performed in these early systems.

In English, temporal expressions are recognised as being highly idiosyncratic, but attempts have been made by linguists to make generalisations of the underlying grammar (Flickinger, 1996). This underlying concept, that the idiosyncrasies can be captured by some generalised rules, is used by automated rule-based annotators.

(Pustejovsky, et al., 2003) identifies three main classifications of temporal expressions:

* “Fully-specified temporal expressions (e.g., June 11, 1989, or Summer 2002);
* Underspecified temporal expressions (e.g., Monday, next month, two years ago);
* Durations (e.g., three months, two years).”

The recognition task is generally considered to be “do-able” (Ahn, Adafre, & de Rijke, Recognizing and Interpreting Temporal Expressions in Open Domain Texts, 2005), with two main approaches to the task: rule-based and machine learning based. Unlike recognition, normalisation is considered a more difficult task, especially for underspecified temporal expressions, and durations.

<<< Normalisation has issues, due to ambiguity, granularity of time, incompleteness of data, etc... Discuss how ambiguity is often handled in different systems. >>>

Becoming frequently common is for the tagger to be incorporated into a larger toolkit that deals with temporal relations <<< such as TARSQI – likely to happen as (Ahn, Adafre, & de Rijke, Recognizing and Interpreting Temporal Expressions in Open Domain Texts, 2005) sensibly suggest separation of roles >>>

<<< Overview of current state of the art for taggers, then introduce individual taggers and the technologies behind them For each one outline system and annotation scheme used

Automated annotation based on hand-written rules

* Mani and Wilson 2000 (hand-written rules for expression tagging) – tempex (basis of GUtime)
* Tarqas (as a component)
* Tarsqi (uses GUTime as a component)
* DANTE
* Chronos
* Saquete
* Mikheev, et al; Krupka, Hausman(early work)

Automated annotation based on machine learning

Ahn, Rantwijck blah includes a particular architecture for machine learning which may be useful

* Mani and Wilson 2000 also includes a discussion of machine learning
* Kolomiyets...
* Hacioglu – SVM
* Baldwin
* Jang, Baldwin, Mani >>>

### GUTime

Talk about GUTime specifically

## Evaluating Tagger Performance

Contests for temporal expression recognition date back as far as the Message Understanding Conference of 1995, but only as part of a broader named entity recognition task. In 2004, the Automated Content Extraction (ACE) programme launched the Time Expression Recognition and Normalization (TERN) evaluation sub-task (MITRE, 2004), which focussed on two sorts of systems – those that do recognition only, and those that do recognition and normalisation.

<<< Talk about TERN (need TERN dataset), TimeBank (have 1.1 – need 1.2), TempEval, AQUAINT. TempEval is for relations, TempEval-2 more interesting >>>

## Rule Discovery

<<< More research needed. Want to look into automated rule discovery Adaptive Information Extraction from Text by Rule Induction and Generalisation >>>

# Project Aims

# Work Plan

## Overview

High level work plan (introduction to different components, etc) about half a page

## Python Tagger

### Overview

Talk about how we’re going to port GUTime, etc

### Workload

|  |  |
| --- | --- |
| Task Name | Estimated Workload |
|  |  |
|  |  |
|  |  |
|  |  |

## Automatic Rule Discovery

### Overview

Talk again (about 2/3rds of a page) about how this is going to work

### Workload

|  |  |
| --- | --- |
| Task Name | Estimated Workload |
|  |  |
|  |  |

## Evaluation

### Overview

Talk about how the evaluation of my system is going to work (basically, as a TERN project)

### Workload

|  |  |
| --- | --- |
| Task Name | Estimated Workload |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Timescale

# Bibliography

Ahn, D., Adafre, S. F., & de Rijke, M. (2005). Recognizing and Interpreting Temporal Expressions in Open Domain Texts. In S. N. Artëmov, H. Barringer, A. S. d'Avila Garcez, L. C. Lamb, & J. Woods (Eds.), *We Will Show Them: Essays in Honour of Dov Gabbay* (Vol. 1, pp. 31-50). College Publications.

Ahn, D., Rantwijk, J. v., & de Rijke, M. (2007). A Cascaded Machine Learning Approach to Interpreting Temporal Expressions. *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics* (pp. 420-427). Rochester, New York, USA: The Association for Computational Linguistics.

Flickinger, D. (1996). English time expressions in an HPSG grammar. *In Studies on the Universality of Constraint-Based Phrase Structure Grammars Gunji* , 1-8.

Krupka, G. R., & Hausman, K. (1998). IsoQuest Inc.: Description of the NetOwlTM Extractor System as Used for MUC-7. *In Proceedings of 7th Message Understanding Conference (MUC-7).*

Mani, I., & Wilson, G. (2000). Robust temporal processing of news. *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics* (pp. 69-76). Morristown, NJ, USA: Association for Computational Linguistics.

Mikheev, A., Grover, C., & Moens, M. (1998). Description of the LTG system used for MUC-7. *In Proceedings of 7th Message Understanding Conference (MUC-7).*

MITRE. (2004). *Time Expression Recognition and Normalization Evaluation*. Retrieved April 28, 2010, from http://fofoca.mitre.org/tern.html

Pustejovsky, J., Castaño, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., et al. (2003). TimeML: Robust Specification of Event and Temporal Expressions in Text. *IWCS-5, Fifth International Workshop on Computational Semantics* .

van Rossum, G. (1995). *Python Reference Manual.* CWI Report CS-R9525.

Verhagen, M. (2004). *Times Between the Lines.* Waltham, MA, USA: Brandeis University.

Verhagen, M., Mani, I., Sauri, R., Knippen, R., Jang, S. B., Littman, J., et al. (2005). Automating temporal annotation with TARSQI. *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions* (pp. 81-84). Ann Arbor, Michigan: Association for Computational Linguistics.