KING'S College LONDON

University of London

6CCS3PRJ Final Year
Automated Timeline Extraction

Final Project Report

Author: Oliver Höhn

Supervisor: Dr Jeroen Keppens

Student ID: 1426248

March 4, 2017

Abstract

The abstract is a very brief summary of the report's contents. It should be about half-a-page long. Somebody unfamiliar with your project should have a good idea of what your work is about by reading the abstract alone. -Summary of Project When legal and related professionals examine a case, they receive a substantial number of documents. These documents need to be examined in a useful manner to understand the events occurred. One useful perspective to understand what happened is a timeline of events. However, reading a large collection of documents and producing a timeline can be cumbersome. The aim is to ease this task by producing a system that can show timelines of events based on a set of documents provided.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Oliver Höhn

March 4, 2017

Acknowledgements

It is usual to thank those individuals who have provided particularly useful assistance, technical or otherwise, during
your project. Your supervisor will obviously be pleased to be acknowledged as he or she will have invested quite a lot of time overseeing your progress.
-Acknowledge Supervisor, Friends & Family I would like to thank my supervisor, Dr. Jeroen Keppens. The supervision and support provided was extremely helpful and helped in the progression of the project. Also I would like to thank my family and friends for the continued support and encouragement throughout the project.

Contents

Chapter 1

Introduction

This is one of the most important components of the report. It should begin with a clear statement of what the project is about so that the nature and scope of the project can be understood by a lay reader. It should summarise everything that you set out to achieve, provide a clear summary of the project's background and relevance to other work, and give pointers to the remaining sections of the report, which will contain the bulk of the technical material. -What is Project About (incl scope)? What is Aim? Bakground? Relevance to other works? (Pointers to other sections?) The project aims to facilitate the understandment of a substantial number of documents, especially in law cases. When an employee is tasked with a law case, it is expected that the fully understand the overall structure and occurrence of events. However, when a large collection of documents are involved, this task can be both cumbersome for the employee and expensive (in time and financially) for the employer. Since time is spent reading and understanding the documents, instead of moving ahead with the task that the documents are used for.

1.1 Project Scope

Due to the system receiving as input a collection of documents, then processing and graphically showing a timeline, the main areas are Natural Language Processing (NLP), and Data Processing and Representation. All of which will be discussed in the Background section.

1.2 Objectives

//what is an event (summary, size, subjects, date, etc) The Objectives are to produce a system that is simple to use and effective. This system should take in as input a selection of documents written in correct English (i.e. in natural language) and produce a timeline.

Since most users are from other fields, not Computer Science, they should be able to understand that the system requires as input documents and produces timelines. The timeline should self-expanatory, such that the user understands which events happened during certain dates, and what each event means. The system should produce responses in an appropriate time based on the input, and allow the user to rectify where the system has made mistakes. In addition, it should allow expert users to be able to change some of the input parameters used when the events are produced, such as the length of the summary or how many processors can the program use in parallel (to limit/improve the performance). As it will be likely that the users will want to save the produced timeline for later use, they will have the ability to choose between saving the timeline as a PDF or as a JSON. The latter allows for the system to be used with 3rd-parties that would like to change the graphical representation or manipulate the given timeline in their own system.

The aim is to develop a system that produces a timeline based on the given input autonomously. The Requirements of the project are:

1. Allow documents of different format types as input (such as .docx, .pdf, .txt).
2. Show a graphical representation of the documents based on the events in them.
3. Allow the editing of produced events, within the constraints of what an event can be.
4. Produce an intemediary output to be saved (as a .pdf or .json).
5. The processing of the system should be reasonable (based on the input size).

1.3 Report Structure

In the following chapter, the background of the project will be presented in detail. Which is then followed by the design architecture and patterns used in the system. Followed by the implementation, testing and analysis of the system. In the analysis it will be determined how the requirements have been met. In the final chapter, the project will be concluded and improvements for future work will be discussed.

Chapter 2

Background

The background should set the project into context by motivating the subject matter and relating it to existing published work. The background will include a critical evaluation of the existing literature in the area in which your project work is based and should lead the reader to understand how your work is motivated by and related to existing work. -the problem with processing text (Twitter experiment)

2.1 NLP

-explain what NLP is -what parts of NLP are involved -cite

2.2 Data Processing and Representation

-explain the event -what are the options for the summary (Neural Networks vs Decision-Based) -give an algorithm for determining the summary, with an example -explain the date problem, with example (determine that it uses an ISO standard)

//defintion of an event An event is given by its date(s), subjects and a short summary of the sentence that produced it. An event can have more than 1 date if it is considered to happen in a range of dates. For example, an event that happened in the 1980s would have two dates, one for the start date: 1980-01-01, and one for the end date: 1989-12-31. While an event that happened just on one day would have just one date. The subjects of an event are given by the "person, place, thing, or idea that is doing or being something" (Grammar.ccc).

Chapter 3

Report Body

The central part of the report usually consists of three or four chapters detailing the technical work undertaken during the project. **The structure of these chapters is highly project dependent**. They can reflect the chronological development of the project, e.g. design, implementation, experimentation, optimisation, evaluation, etc (although this is not always the best approach). However you choose to structure this part of the report, you should make it clear how you arrived at your chosen approach in preference to other alternatives. In terms of the software that you produce, you should describe and justify the design of your programs at some high level, e.g. using OMT, Z, VDL, etc., and you should document any interesting problems with, or features of, your implementation. Integration and testing are also important to discuss in some cases. You may include fragments of your source code in the main body of the report to illustrate points; the full source code is included in an appendix to your written report. -Tasks in project (Design, Implementation, Experimentation, Optimissatio, Evaluation) -present alternatives, compare them, why picked -justify software used -problems identified -important features -testing

3.1 Section Heading

3.1.1 Subsection Heading

Chapter 4

Design & Specification

-design of architecture and UI

4.1 Section Heading

Chapter 5

Implementation

-How implemented

5.1 Section Heading

Chapter 6

Professional and Ethical Issues

Either in a seperate section or throughout the report demonstrate that you are aware of the **Code of Conduct & Code of Good Practice** issued by the British Computer Society and have applied their principles, where appropriate, as you carried out your project. -how dealt with ethical approval? (newspapers used, etc.) -in analysis kept testers anonymous

6.1 Section Heading

Chapter 7

Results/Evaluation

-present how did analysis, why?, other options (relate to work) -results of analysis -conclusion

7.1 Software Testing

7.2 Section Heading

Chapter 8

Conclusion and Future Work

The project's conclusions should list the key things that have been learnt as a consequence of engaging in your project work. For example, "The use of overloading in C++ provides a very elegant mechanism for transparent parallelisation of sequential programs", or "The overheads of linear-time n-body algorithms makes them computationally less efficient than $O(n \log n)$ algorithms for systems with less than 100000 particles". Avoid tedious personal reflections like "I learned a lot about C++ programming...", or "Simulating colliding galaxies can be real fun...". It is common to finish the report by listing ways in which the project can be taken further. This might, for example, be a plan for turning a piece of software or hardware into a marketable product, or a set of ideas for possibly turning your project into an MPhil or PhD.

-what have you learned? -how can the project be carried further (neural net for summary, building on the StanfordCoreNLP for detas depending on others)

Appendix A

Extra Information

A.1 Tables, proofs, graphs, test cases, ...

The appendices contain information that is peripheral to the main body of the report. Information typically included in the Appendix are things like tables, proofs, graphs, test cases or any other material that would break up the theme of the text if it appeared in the body of the report. It is necessary to include your source code listings in an appendix that is separate from the body of your written report (see the information on Program Listings below).

Appendix B

User Guide

B.1 Instructions

You must provide an adequate user guide for your software. The guide should provide easily understood instructions on how to use your software. A particularly useful approach is to treat the user guide as a walk-through of a typical session, or set of sessions, which collectively display all of the features of your package. Technical details of how the package works are rarely required. Keep the guide concise and simple. The extensive use of diagrams, illustrating the package in action, can often be particularly helpful. The user guide is sometimes included as a chapter in the main body of the report, but is often better included in an appendix to the main report. -how to set up (commands) -how to use given pieces of sample text -images

Appendix C

Source Code

C.1 Instructions

Complete source code listings must be submitted as an appendix to the report. The project source codes are usually spread out over several files/units. You should try to help the reader to navigate through your source code by providing a "table of contents" (titles of these files/units and one line descriptions). The first page of the program listings folder must contain the following statement certifying the work as your own: "I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary". Your (typed) signature and the date should follow this statement.

All work on programs must stop once the code is submitted to KEATS. You are required to keep safely several copies of this version of the program and you must use one of these copies in the project examination. Your examiners may ask to see the last-modified dates of your program files, and may ask you to demonstrate that the program files you use in the project examination are identical to the program files you have uploaded to KEATS. Any attempt to demonstrate code that is not included in your submitted source listings is an attempt to cheat; any such attempt will be reported to the KCL Misconduct Committee.

You may find it easier to firstly generate a PDF of your source code using a text editor and then merge it to the end of your report. There are many free tools available that allow you to merge PDF files.