Project Proposal

Computer Science Tripos, Part II

A Study Helper based on Natural Language Processing

L.A. Bura, Trinity College

20 October 2017

Project Originators: Ekaterina Kochmar & Lucia Bura

Project Supervisor: Ekaterina Kochmar

Directors of Studies: Dr Sean B. Holden & Dr Arthur C. Norman

Project Overseers: Prof. Marcelo Fiore & Dr Timothy Jones

Introduction

The aim of my project is to build an educational tool based on **key–phrase extraction**, **summarization** and **question generation** algorithms. This tool would extract the main information from a given text fragment/document and proceed to generate questions concerning those points. It could then be used to test factual knowledge of a text which would ultimately help learners understand the material or prepare for examinations. The domain in which it would be used is academic, more precisely lecture notes and papers on computer science.

This project will be divided into three main parts, tightly connected, all of which are techniques introduced by the **natural language processing** (NLP) area of computer science.

→ Firstly, we will use key–phrase extraction on the text, that is, automatically identify a set of terms that best describe the given material.

→ Next, we will use this in conjuction with **single-document**, **extractive** summarization. Extractive in this context simply means we will divide the sentences into different groups based on those keywords, then choose the ones which seem to be most illustrative of the content as a whole by analyzing them individually and assigning a score. The one with the higher score is the winner of its cluster. This implies high similarity between extracted sentences and existing ones in the original text, which is good in terms of the academic material to be used, as it generally requires learning specific scientific terms.

→ The final step is taking these sentences and generating **factual** questions. This will be done by performing modifications on the sentence structure and turning it into a query–like construction, using classical NLP techniques.

Ultimately, the goal of this application is to aid students in their learning attempts. Using the approaches outlined above I hope to achieve results of a comparably better quality than existing summarization and question generation tools on the same set of data.

Starting Point

The implementation of my project will rely on previous programming experience, a few existing software libraries as well as knowledge drawn from courses on the Computer Science Tripos.

* **CST COURSES:**
  - → ***Natural Language Processing (Part II):*** My project will be mostly based on the techniques and ideas outlined in this year's overview of NLP course. It will build upon the more classical methods of sentence processing and span across multiple topics covered in lectures, aiming to elaborate on keyword extraction, summarization and language generation. Evaluation methods mentioned in the course will also be employed.
  - → ***Artificial Intelligence I (Part IB)***
    & Machine Learning and Bayesian Inference (Part II): AI I gives a good introduction to the underlying mathematical background as well as fundamentals of artificial intelligence. This, combined with the Part II continuation of the course define a set of algorithms and tools to be employed and experimented with in my project. Unsupervised learning methods, clustering in particular, are especially of interest as potential candidates for the core of my project. Concepts captured later regarding recurrent neural networks are also considered for potential extensions envisioned.
  - → ***Information Retrieval (Part II):*** To evaluate the project and understand the relevance of my results, techniques outlined in the IR course this year will be used.

NLP, Machine Learning and Bayesian Inference, Information Retrieval will be undertaken this year, two of which only start in Lent term. This implies I will be familiarising myself in advance with the contents of these courses. Further material is therefore to be consulted, in the form of papers, open online courses as well as course textbooks.

* **OPEN SOURCE LIBRARIES:**
  - → ***NLTK:*** A very useful platform for interacting with human language data. It's a collection of Python libraries that come with support for several text processing actions including classification, tokenization, PoS tagging. I will employ NLTK and its accompanying book for preprocessing and parsing the text extracts.
  - → ***TensorFlow:*** Very useful open-source software library built by Google for numerical computation using data flow graphs. This may prove valuable for the machine learning efforts in my project. Using the Python API of `TensorFlow` I will be able to experiment with the appropriate algorithms for the project extenstions.

* **PROGRAMMING EXPERIENCE:**
  - → ***Python:*** I have experience programming in Python from the last two summer internships I've done. This included making use of Keras for the basic training and testing of a *Long Short Term Memory* (`LSTM`) recurrent neural network architecture.

Work to be done

The primary goal of this project is to summarize the key points of a given document and present them in the form of natural language questions. These are meant to assess factual knowledge of most important bits of information.

To achieve this goal, I have structured my work in 4 blocks of systematic progress, as follows:

→ **Preparation** – In order to prepare for both implementation and evaluation, background reading in existing NLP techniques for summarization, keyword extraction and query generation is required. Finally, experimentation with the open-source libraries and frameworks (`NLTK`, `scikit-learn`) that are available for the task should be undertaken. To set a performance benchmark, existing automatic summarization tools (`Baseline`, `TextRank`, etc) will be experimented with.

→ **Core project implementation** – The following main components are necessary to achieve a functioning educational tool:

- **Implement the keyword extraction algorithm**. This will be done using unsupervised learning methods similar to the ones employed in TextRank [1].
- **Implement the summarization algorithm**. Basic NLP sentence extraction and word-frequency counts will be used in conjunction with the above step. Clustering and rule-based approaches will ensure key sentence diversity in a larger document.
- **Implement the question generation algorithm**. This last step is going to consist of two smaller steps, one will be the generation of a few candidate questions using classical NLP approaches outlined in Heilman [2]. Salient portions of sentences will already have been provided by the key–phrases previously extracted, smoothing the process. These questions will then be ranked and the best one will be chosen.

The main implementation language will be Python. Classical NLP techniques required for text processing will be implemented using the `NLTK` platform. The three–step pipeline outlined above will have each of its main components individually tested.

→ **Core project evaluation** – To evaluate the performance of the system, I will consider:

- **Keyword relevance** – F1–score to measure amount of relevant information contained. That is, a harmonic mean of *precision* (the proportion of extracted key-phrases that match the expert assigned key-phrases) and *recall* (the proportion of expert key-phrases that made it into our selection)
- **Summary relevance** – ROUGE toolkit *(Recall-Oriented Understudy for Gisting Evaluation)* to measure units such as n-gram, word sequences, and word pairs between system–generated and reference summaries.
- **Generated question correctness** – Grammatical correctness will be measured, using unlexicalised PCFG parsers, as suggested in a recent paper [3].

Evaluation will be performed on a corpus of annotated computer science papers and lecture notes with summaries available online [4]. I will also run the tool against Cambridge Computer Science Tripos lecture notes demonstratively.

→ **Dissertation** – A clear explanation of the background material, steps taken and the achieved results is required, after successfully conducting the above steps.

Success Criteria

To be considered a successful project, **Preparation, Implementation, Evaluation** and **Dissertation** explained above have to be *completed* and *well-documented*. The end result should be a demonstratable tool, producing questions based on the main sentences of a given text.

Textual clarity is very important for NLP applications, therefore scores produced after checking for grammatical correctness in generated questions should reflect that.

Another success criterion is for my tool to perform at least as good as the unspecialised, established models after conducting a performance comparison.

Comparisons and evaluations will be mostly quantitative therefore plots and tables will provide a nice visualisation, their presence in the final dissertation document being a success in its own.

Possible Extensions

Should the core of my project be finalised according to plan, I will try the following extensions:

1. **Implement and evaluate a topic-based summarization** – Optimizing the existing summarization algorithm by allowing the user to specify a topic of interest. This would lead to questions related to the input topic. For instance lecture notes on networking with the additional topic: *routing* would result in questions more routing–oriented.
2. **Implement and evaluate a multi-sentence compression algorithm** – (i.e throwing away irrelevant bits of information) in order to optimize the summarization algorithm further. This would be attempting an abstractive approach to summarization.
3. **Question generation using machine learning methods** – using a different approach to query generation machine learning approaches proposed in literature in connection with the produced key–phrases. For this extension I would use `TensorFlow` open-source library and the Python `scikit-learn` library.

Resources Required

I plan to use my personal laptop, an Asus laptop with a 2.4 GHz Intel Core i7 CPU, 12GB RAM and a 128 GB SSD running Windows 10 and Ubuntu 16.04 in dual-boot, as the main machine for doing the implementation, evaluation and dissertation write-up. I will also be employing relevant open-source frameworks and libraries such as `NLTK`, `TensorFlow`, `scikit-learn` as well as software for testing and source controlling.

I will use following contingency plans to protect myself against hardware/software failures:

- Revision control for the code and the LaTeX dissertation write–up using `git` by regularly pushing the code into a repository on GitHub.
- Performing regular backups of entire project files and folders to an external 1 TB HDD

Having taken these precautions, should my personal laptop suddenly fail, I will be able to continue working on a MCS machine. I require no other special resources.

Timetable and Milestones

The main timeline goal is to complete the core project by mid December at its latest. I will therefore be able to dedicate my time to implementing proposed extensions by early February. Finally, I should produce a final dissertation write–up ready for revision and submission by mid-April. I have divided my time roughly in two–week periods with corresponding milestones and deadlines when applicable, as follows:

→ *5th October – 19th October*

- ∼ Working on the project proposal.
- ∼ Setup of all contingency schemes as described above

Milestone: Have project proposal ready for submission.
Deadline: Phase 1 Report (9 Oct), Draft Proposal (13 Oct)

→ *20th October – 3rd November*

- ∼ Familiarise myself with classical approaches and evaluation metrics for summarization and keyword extraction by reading relevant literature.
- ∼ Extract main textual content from online sources and course notes pdfs
- ∼ Experiment with established summarization tools on these and check performance
- ∼ Become more acquainted with `NLTK`, `TensorFlow` and `scikit-learn` libraries

Milestones: Finished preparatory background reading. Have main corpora ready
[PREPARATION COMPLETED]
Deadline: Project proposal submission (20 Oct)

→ *4th November – 16th November*

- ∼ Create project workspace and application infrastructure.
- ∼ Implement and evaluate classical keyword extraction.

Milestones: Project space ready. Implemented and evaluated basic keyword extraction.

→ *17th November – 30th November*

- ∼ Implement and evaluate algorithm for extractive summarization on text fragment.
- ∼ Add unit tests for keyword extraction

Milestones: Implemented and evaluated summarization algorithm. Added unit tests.

→ *1st December – 17th December*

- ∼ Implement and evaluate question generation algorithm
- ∼ Add unit tests for sentence extraction and question generation

Milestones: Question generation algorithm done. End-to-end system working.
[CORE IMPLEMENTATION AND EVALUATION FINISHED]

→ *18th December – 5th January*

- ∼ Buffer slot for slack.
- ∼ Implement and evaluate topic-based summarization.
- ∼ Draft progress report and send to supervisor for feedback.

Milestones: Draft of progress report finished. Implemented and evaluated topic–based summarization. ***[1/3 EXTENSIONS FINISHED]***

→ *6th January – 23rd January*

- ∼ Implement multi-sentence compression.
- ∼ Add unit tests.

Milestone: Implemented multi-sentence compression.
[2/3 EXTENSIONS FINISHED]

→ *24th January – 2nd February*

- ∼ Read relevant literature on machine learning approaches to question generation.
- ∼ Prepare training data.
- ∼ Prepare slides and material for progress report presentation

Milestones: Training data ready. Slides ready. Delivered progress report.
DEADLINE: Progress report due on 2nd Feb *(12 noon)*

→ *3rd February – 20th February*

- ∼ Implement machine learning question generation algorithm.
- ∼ Integrate it with summarization for more accurate key–phrase question scoring

Milestones: Implemented question generation algorithm. Delivered progress report presentation.
DEADLINE: Progress report presentation *(sometime on 8 – 13 Feb)*

→ *21st February – 4th March*

- ∼ Optimize and evaluate question generation algorithm.

Milestones: Prepared for presentation. Evaluated question generation algorithm
[3/3 EXTENSIONS FINISHED]

→ *5th March – 12th March*

- ∼ Buffer slot for slack.
- ∼ Add more unit tests where appropriate.
- ∼ If everything went according to plan, start working on dissertation early.

Milestones: All tests are in place. Ready for write–up.

→ *13th March – 25th March*

∼ Work on dissertation, complete **Introduction** and part of **Implementation** – Preparation

Milestones: Completed Introduction and Preparation chapters.

→ *26th March – 4th April*

∼ Work on dissertation, complete **Implementation** and **Evaluation**

Milestones: Completed Implementation and Evaluation chapters.

→ *5th April – 12th April*

∼ Proofread and prepare dissertation for first draft submission.

∼ Submit first draft to supervisor and Directors of Studies for feedback.

Milestone: Submitted first draft of dissertation.

→ *13th April – 20th April*

∼ Amend dissertation for second draft submission.

∼ Submit second draft to supervisor and Directors of Studies for feedback.

Milestone: Submitted second draft of dissertation.

→ *21st April – 28th April*

∼ Amend dissertation after feedback.

Milestone: Dissertation ready for submission.
Ultimate Deadline: 18 May 2018 *(12 noon)*

References

[1] Rada Mihalcea and Paul Tarau. *TextRank: Bringing Order into Texts*. Association for Computational Linguistics (2004).

[2] Michael Heilman. *Automatic factual question generation from text*. Carnegie Mellon University Pittsburgh, PA, USA 2011

[3] Alexander Gamble (2017) *Improving and Evaluating Methods for Automatic Factual Question Generation*. Cambridge University

[4] Krapivin, Mikalai and Autaeu, Aliaksandr and Marchese, Maurizio (2009) *Large Dataset for Keyphrases Extraction*.