UNIVERSITY OF BIRMINGHAM

Final Year Project

Extracting Key Phrases and Relations from Scientific Publications

Dissertation for B.Sc in Computer Science

School of Computer Science, University of Birmingham

Author
Thomas Clarke (1443652)

Supervisor
Dr Mark Lee

April 2018

Declaration

The material contained within this thesis has not previously been submitted for a degree at the University of Birmingham or any other university. The research reported within this thesis has been conducted by the author unless indicated otherwise.

Acknowledgements

I would like to give acknowledgement to those who helped me throughout the completion of this project.

Firstly, a thank you to Dr Mark Lee for being a supportive and informative supervisor, as well as an entertaining host during project meetings.

I also wish to thank my friends and family in supporting me during the year leading preceding this dissertation, ensuring I kept on track and in a good frame of mind.

Abstract

This project presents solutions developed to solve the SemEval 2017 ScienceIE task - analysis of scientific publications to extract key information. This includes three sub tasks: *(A)* key phrase extraction, *(B)* classification and *(C)* relation extraction.

To achieve subtask A, the text of a paper is parsed to find it's semantic tree. Then, each word in succession is tested in a Support Vector Machine (SVM), based around a words' semantic attributes to determine if it should be a, or part of a, key phrase. Each phrase generated is also sanitised to reduce excess information. Subtask B involved treating each key phrase as a Bag-Of-Words, and calculating the phrases' distance to each classification type using Word2Vec. Finally, subtask C experimented with using the Word2Vec representation of a phrase and the relative distances between phrases combined with an SVM to try too detect relations.

Scores of NLP go here

To explore how this system could be used, a website was created hosting the information. This used Spring Boot to create a Java based web project which supported not only an archive of processed papers, but also the means to search using query strings and automatic processing of submitted papers to the system (through using the most successful versions of systems described above). *Evaluation summary goes here...*

Keywords

Natural Language Processing, Key Phrase Extraction, Classification, Relation Extraction, Support Vector Machine, Word2Vec, Spring Boot

Contents

List of Figures

List of Tables

1 Introduction

When conducting scientific study, being able to search existing literature around a subject can be vitally important. A search system which can automatically sort scientific papers into order, returning the one likely to be most useful first, can speed up the process of gathering this information. A system which can go further and extract important pieces of information from the paper to help present answers to user queries has the potential to be even more effective.

At SemEval 2017[1], a task which heavily applied to the above was presented: ScienceIE[2]. This natural language processing (NLP) based task was to analyse scientific papers to extract key pieces of information, classify those pieces and attempt to draw relations between them. In short, this is a information extraction problem, specifically for scientific papers. The idea behind it is to support faster research as systems will be presented with information to help better gather relevant research when querying databases of existing literature.

1.1 Aims and Objectives

This project shall initially target the main goals of ScienceIE, and one of the methods of evaluation shall be through processing of the sample data and execution of the marking tools supplied as part of the task. Explicity, the overall task is split into 3 subtasks:

- **A**: The identification of all the key phrases in a scientific publication
- **B**: The classification of each key phrase into one of the following categories:
  - **Process** (scientific models, algorithms, processes)
  - **Task** (an application, end goal, problem, task)
  - **Material** (resources, materials)
- **C**: The identification of relationships between identified key phrases, where the relation is either none, or one of the following:
  - **Hyponym-of** (where the semantic field of key phrase A is included in that of key phrase B's semantic field, but not vice versa)
  - **Synonym-of** (where the semantic field of key phrase A and B are the same)

Therefore, through research of the systems created during ScienceIE and other research in the field, the largest and most obvious goal of this project is to create a system where any scientiific paper can be input, some processing happens (with no time constraints) and the desired key phrase information is produced as an output, in the expected format specified for ScienceIE. This is the 'brat' annotations format, which houses all of information described above about a paper in a single text document, saved separately from the original paper.

ScienceIE have supplied sample development, training and testing data for use in those participating in the task. An example *paper* can be see in appendix A and the annotations file that goes with it can be seen in appendix B.

The above can be refereed to as the *NLP system* part of the project. To evaluate the NLP system, not only will the marking tools be used, but further analysis of the information extracted shall also be conducted; for instance exploring the differences in key phrases automatically extracted compared to the expected results (seeing cases where a shorter or longer key phrase was extracted and what difference this might make when using the generated data).

There currently existing many search engines that specifically deal with research papers; Google Scholar[3] and ScienceDirect[4] are well known, popular choices currently. As an extension to the ScienceIE task,

[1] http://alt.qcri.org/semeval2017/
[2] https://scienceie.github.io/
[3] https://scholar.google.co.uk/
[4] https://www.sciencedirect.com/

motivated by existing search engines publicly available on the web, the secondary goal of this project is to create a *proof-of-concept (POC) product* based on the NLP system. It should use information extracted by the NLP system to present useful information to the user, given suitable input through a graphical user interface (GUI).

It should maintain a collection of scientific papers that are prepared for user query to effectively help them navigate to the most useful piece of information relating to their query first. As a minimum requirement, it should host at least the test data supplied by ScienceIE. The papers should be able to be read in full, or simply have the extracted information presented (at least in the brat format described above) for the users convenience.

The goal of this *POC system* section of the project is to be able to explore the potential effectiveness of the extracted information in relation to a researcher trying to find relevant research and how effectively it can be presented to aid in understanding it.

1.2 Report Outline

This document begins with a background to the field, which feeds into specification of what is to be explored and implemented. This will cover the NLP system in detail and outline the GUI requirements, as this shall be discussed more towards the latter parts of the paper. Following that, the NLP system implementation is reported on, concluded in the next section which evaluates the strengths and weaknesses of the NLP system. Once the NLP system has been discussed, the POC concepts shall be explained in full, the implementation discussed and evaluation completed. To sum up, a final discussion section shall review the project as a whole, and reiterate the strongest positives and note some of the points for improvement or expansion.

2 Background and Literature Review

2.1 ScienceIE Proceedings

Evaluating the outcome of ScienceIE at SemEval indicates potential paths for future systems and document very recent activity in the key phrase extraction area. Three papers were published from the event regarding this task.

Firstly, an overview of how successful the task was shall be conducted. The highest end-to-end F1 score achieved by any team was measured to be 0.43 for all three sub-systems combined (Augenstein, Das, Riedel, Vikraman & McCallum 2017), with each of sub tasks A, B and C were 0.56, 0.44 and 0.28 respectively.

For sub task A, it was evaluated that while many high scores were achieved with recurrent neural networks (NN), the highest scoring system was a support vector machine (SVM) using a well-engineered lexical feature set. SVMs and NNs were also popular choices for sub task B. For sub task C, many methods were attempted and while a convolutional NN was the most effective, various other methods (including SVM and multinomial naïve Bayes) all achieved very similar and reasonably accurate scores (the best had an F1 score of 0.64 when evaluated solely on sub task C).

– Needs rewriting after this bit –

The best end-to-end ScienceIE team used a long short-term memory (LSTM) approach for phrase extraction, with labelling completed by a conditional random field (CRF) based sequence tagging model (Ammar, Peters, Bhagavatula & Power 2017). Their sequence tagging model employed gazetteers built from scientific words extracted from the web. Another team (Marsi, Sikdar, Marco, Barik & Sætre 2017) also had similar ideas, using CRFs to complete some of the task using WordNet[5] as a data source for the classifier they created. Both teams here also used sensible rules to help improve their score, such as intuitively marking all instances of a key phrase as a key phrase upon finding one instance (so if 'carbon' is extracted and labelled as a 'material', then all other instances are labelled to match) and exploiting hypernym relationship's bidirectional property (so if word 1 is a hypernym of word 2, the reverse is also true and therefore recorded).

Unfortunately, while extraction and classification were generally well handled, relation extraction has very low accuracy across all teams taking part with the average F1 score only being 0.15, with the highest score being 0.28. (Ammar et al. 2017) achieved this using gazetteer built from Wikipedia[6] and freebase. This seems more appropriate than hand written rules, which may seem appealing as they can be somewhat tailored and provide high accuracy, but require much more effort from the developer and may not work well on unseen conditions providing low accuracy (Manning & Jurafsky 2012). As mentioned earlier, WordNet is also a potential source of information for building a classifier for relation extraction, and a study by (Snow, Jurafsky & Y. Ng 2013) compared building a classifier off of Wordnet and Wikipedia for hypernym-only extraction. The result of this shows that Wikipedia may be more suited to creating this type of classifier as it achieved an F1 score higher than using WordNet (the Wikipedia based classifier got 0.36 while the WordNet based classifier got 0.27). While an improvement, it is not ultimately a huge increase and there is no evidence either Wikipedia is better for the specific area of scientific papers (as the 2013 study was completed on a generic set of data).

The results of ScienceIE demonstrate there are several potential systems that could be implemented to answer this problem, with the best system potentially being a combination of algorithms and a voting system to select and label key phrases. The product would likely involve supervised learning and previous knowledge for some algorithms, along with unsupervised learning sections as well.

2.2 Other Revelant Background Information

Several teams from ScienceIE chose to back up key phrase extraction with a CRF. CRFs can be used for key phrase extraction alone as well (Zhang, Wang, Liu, Wu, Liao & Wang 2008), and while studies imply that CRFs (shown to have F1 scores of 0.51) are more accurate than an SVM (the most accurate at ScienceIE) this paper is slightly older than papers produced at SemEval 2017 and so even if the SVM information used then was the best that was available at the time (the SVM F1 score was 0.46), the SVM implemented at ScienceIE beat both of these scores considerably achieving an F1 of 0.56 as mentioned above.

[5] https://wordnet.princeton.edu/

[6] https://en.wikipedia.org

	Minimum	Average	Maximum	Standard Deviation
Nnumber of KPs	4	19	46	8
Minimum tokens per KP	1	1	3	0.4
Average tokens per KP	1	3	8	1
Maximum tokens per KP	2	9	25	4
Number of relations	0	2	13	2
Total tokens in document	60	159	264	46

Table 1: Key phrase (KP), token and relation analysis for the ScienceIE training set.

A method not attempted at ScienceIE was unsupervised learning by clustering key phrases, a method which has potentially very accurate results that also could not only be robust again new unseen data but even different languages. The idea is that candidate key phrases are selected by some heuristic and other phrases are clustered about them. With the simplest approach, the center of a cluster is the key phrase. Various clustering methods were attempted by (Liu, Li, Zheng & Sun 2009) on top of a candidate selection process built on semantic term relatedness. They ran tests on relatively short articles and while at maximum they only achieved an F1 of 0.45, there was several improvements suggested which apply to the task at hand concerning scientific papers. Firstly, an achievable improvement for this project would be to cluster directly on noun groups as they found most clusters consisted of groups of nouns anyway, which is backed up by Augenstein et al. (Augenstein et al. 2017), who reports 93% of all key phrases are noun phrases. Furthermore, improving their initial filtering to extend it further than stop words may help reduce errors as well; improving this may be possible by employing a words TF-IDF score with some threshold. Finally, they suggested a similar algorithm be applied to longer scientific papers. ScienceIE's test data consists of extracts of scientific texts (i.e. short paragraphs), however, any unsupervised system created for this task could be ran again entire papers and then only those sections compared for evaluation later – allowing this suggestion to be evaluated.

2.3 The Supplied Data Set

The ScienceIE data set consists of 50 development, 350 training and 100 test documents.

Some analysis conducted at ScienceIE (Augenstein et al. 2017) showed some characteristics of the sample key phrases included:

- Only 22% of key phrases had 5 or more tokens,
- 93% of key phrases were noun phrases,
- Only 31% of key phrases seen in the training set were also in the test set.

This means that key phrase extraction appears quite difficult, as an algorithm needs to search for short phrases, processing phrases that it likely hasn't seen instances of before. Most of the key phrases being noun phrases, however, is valuable information as it helps to identify a simple heuristic that can be used when processing.

Other useful and interesting characteristics about the training set, found during this study, can be seen in table 1.

Papers in the ScienceIE data set have many key phrases associated with them. With an average of 19 key phrases per paper, an average of 3 tokens per key phrase (meaning on average 57 key tokens per paper) and the average document containing only 159 tokens in total, around a third of all tokens are part of key phrases. This is partly due to the documents supplied by ScienceIE being very short (all are just one paragraph) and are *extracts* of papers rather than full publications. It is not a problem that the documents for processing are short - in fact that may help as the longer the document, the harder it is to choose key phrases (Hasan & Ng 2014) - however, it may mean any algorithm created here may not scale well to full scientific papers. It seems the ScienceIE task is looking for localised key phrases, choosing several from one paragraph; while the author of a paper may choose to select just five or ten key phrases from the whole paper. While this project will focus on the ScienceIE task with the given test data, a brief look longer or full papers shall be considered.

3 Analysis and Specification

Say what I'm going to do, but probably a bad idea to have a section for this. It may work better to just have a all of the 'what im doing' in each section when we get there.

3.1 Project Architecture

With any large software project, it is sensible to choose a platform with all the necessary tools available so the developer can achieve their goals.

Due to the past experience of the author, Java was an obvious choice. Given extensive time working in the language during university and in industry, a thorough understanding of the programming language was already achieved, which allowed for planning of a sensible software architecture to optimise code quality and (implicitly) the potential of increased success of the systems created.

Furthermore, Java is a very popular and accessible language world wide - backed up by the active StackOverflow community (casual and professional alike) with Java being one of the most popular technologies for at least the last five years, evidenced through their user surveys 2018[7], 2017[8] and 2016[9]. Due to this, Java has extensive support for many common problems people encounter, with issues being discussed and solutions proved across various forums.

Not only is Java's popularity good for increasing support availability, many libraries and utilities are available to help developers with tasks. Along side other technologies used for more specific tasks throughout completion of the NLP system and the POC system (which shall be discussed when used), common technologies used during the development of the entire project are described below. Throughout development of the project, very little issue was caused by lack of Java support for common processes or lack of Java capability when attempting to program some process.

log4j

log4j 2[10] is a popular and robust library developed under the Apache Software Foundation to do logging in Java. It's useful features include:

- Automatic output of logs to both terminal and file: As well as immediate visual feedback, log files can be used for later processing and evidence gathering.
- Timing of events: Timing is very useful as during long runs of a system (for example, some sections of the NLP task could take hours to complete) the logs can be analysed to see how long systems take to process data, which can be considered when going forward; for instance in terms of formulating efficiently timed tests plans.
- Labelling of logs into levels such as 'debug', 'info', 'error' and 'fatal' messages: This can be used when analysing the logs to catch where things went wrong (filtering for error messages) and then to try to debug the system by finding information logged prior to that (with debug). During development an excellent use of this feature is to output all levels aside from 'debug' to terminal, so monitoring progress isn't overloading the executor with information, but if something does go awry the steps leading up to the bad event can be analysed in the log saved to disk.

While direct output of this will not be present in the rest of this report, it is worth noting this was an extremely useful tool for developing all of the systems to follow.

Maven

Apache Maven[11] is another important tool. Like log4j, it is developed by the Apache foundation.

[7] https://insights.stackoverflow.com/survey/2018
[8] https://insights.stackoverflow.com/survey/2017
[9] https://insights.stackoverflow.com/survey/2016
[10] https://logging.apache.org/log4j/2.x/
[11] https://maven.apache.org/

Maven is a tool to help with project management and has many uses. It is based around a 'project object model' (POM) configured in a `pom.xml` file at the root of a Java project, which itself has a structure defined by Maven. The key uses utilised in this project are:

- Project compilation: Maven can be used to build a project and automatically run specified or all tests, with more detailed and well formatted output than compiling Java code by hand. Therefore, compilation and testing can more easily be scripted and output more clearly analysed. It also handles importing libraries used in a Java project when compiling (which can be very troublesome when completed by hand), which is discussed below.

- Library import: The `pom.xml` can specify dependencies of the Java project. While custom, third party repositories exist, Maven has a central repository[12] with many libraries available. This includes log4j described above, and all other libraries used in this project. Dependencies are downloaded to the systems local Maven repository at compile time.

- Library export: As discussed in the introduction, the NLP system shall be used in a POC system. Rather than combining these two systems into one large package, or doing a confusing copy of the required resources, Maven can be used to export the compiled NLP system to the local Maven repository. Then, the POC system can simply list the NLP system as a dependency, and Maven shall include it as a library when building the executable program.

Maven is used as the management backbone throughout the development of software discussed in this report. When libraries are used in a project, a link to their dependency configuration for Maven's `pom.xml` shall be included. As a good example, log4j[13] has an extensive page providing a detailed description of how to import the library.

[12] http://repo.maven.apache.org/maven2/
[13] https://logging.apache.org/log4j/2.x/maven-artifacts.html

4 The ScienceIE Task: Design and Implementation

To complete the ScienceIE task, three systems would need to be created to handle each sub task.

4.1 Data Preprocessing

4.2 Subtask A - Key Phrase Extraction

A section all about what I did for part 1

4.2.1 Method 1: Support Vector Machine

Go through making the SVM and what tests helped a lot. As part 2 has already been described, I think it makes sense here to mention I tried adapting this slightly for task 2 but that it went terribly.

4.2.2 Method 2: Clustering

Talk about the experimentation with clustering.

4.3 ScienceIE Subtask B - Key Phrase Classification

A section all about what I did for part 2

4.3.1 Word2Vec Classification

Talk about using word2vec to simply find a good way to quickly classify key phrases with decent results. *** Where do I fit the SVM for this, as not worth a whole section

4.4 ScienceIE Subtask C - Relation Extraction

A section all about what I did for part 3

4.4.1 Support Vector Machine

Discuss the SVM I tried to do this with (including Word2Vec)

5 The ScienceIE Task: Evaluation

How each section went, including test results and maybe some info on other experiments.

5.1 Subtask A - Key Phrase Extraction

5.1.1 Conclusion

5.2 ScienceIE Subtask B - Key Phrase Classification

5.2.1 Conclusion

5.3 ScienceIE Subtask C - Relation Extraction

5.3.1 Conclusion

6 Creating a Proof of Concept Use for ScienceIE data

Having completed systems to handle the information extraction, the next major part of the project was to explore using this information in a generally useful way.

6.1 Further Research

Discuss the resources used to design maybe? Make sure to include research on searching I did...

6.2 Design and Implementation

How it was pulled off

6.3 Web Interface

Exactly what was achieved

6.4 Testing

(Get) user feedback

6.5 Conclusion

Overall impact of the GUI on the project

7 Discussion

Talk about overall results

7.1 Improvements and Extensions

References

Ammar, W., Peters, M., Bhagavatula, C. & Power, R. (2017), 'The AI2 system at SemEval-2017 Task 10 (ScienceIE): semi-supervised end-to-end entity and relation extraction', *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* **10**, 592–596.
URL: *http://www.aclweb.org/anthology/S17-2097*

Augenstein, I., Das, M., Riedel, S., Vikraman, L. & McCallum, A. (2017), 'SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications', pp. 546–555.
URL: *http://arxiv.org/abs/1704.02853*

Hasan, K. S. & Ng, V. (2014), 'Automatic Keyphrase Extraction: A Survey of the State of the Art', *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* pp. 1262–1273.
URL: *http://aclweb.org/anthology/P14-1119*

Liu, Z., Li, P., Zheng, Y. & Sun, M. (2009), 'Clustering to Find Exemplar Terms for Keyphrase Extraction', *Language* **1**, 257–266.
URL: *http://portal.acm.org/citation.cfm?doid=1699510.1699544*

Manning, C. & Jurafsky, D. (2012), 'Using Patterns to Extract Relations'.
URL: *https://www.youtube.com/watch?v=VodeEgvxgtA*

Marsi, E., Sikdar, U. K., Marco, C., Barik, B. & Sætre, R. (2017), 'NTNU-1$@$ScienceIE at SemEval-2017 Task 10: Identifying and Labelling Keyphrases with Conditional Random Fields', *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* pp. 937–940.
URL: *http://www.aclweb.org/anthology/S17-2162*

Snow, R., Jurafsky, D. & Y. Ng, A. (2013), 'Learning syntactic patterns for automatic hypernym discovery', *Journal of the American Medical Informatics Association* **20**(1), 1–11.
URL: *http://dx.doi.org/10.1186/s12859-015-0606-0%5Cnhttp://dx.doi.org/10.1016/j.jbi.2015.02.004%5Cnhttp://dx.do*

Zhang, C., Wang, H., Liu, Y., Wu, D., Liao, Y. & Wang, B. (2008), 'Automatic Keyword Extraction from Documents Using Conditional Random Fields', *Journal of Computational Information* **43**, 1169–1180.
URL: *http://www.jofci.org*

A Example ScienceIE Training/Test Document

The following is ScienceIE test paper file S0010938X15301268.txt:

Fig. 9 displays the growth of two of the main corrosion products that develop or form on the surface of Cu40Zn with time, hydrozincite (Fig. 9a) and Cu2O (Fig. 9b). It should be remembered that both phases were present already from start of the exposure. The data is presented in absorbance units and allows comparisons to be made of the amounts of each species between the two Cu40Zn surfaces investigated, DP and HZ7. The tendency is very clear that the formation rates of both hydrozincite and cuprite are quite suppressed for Cu40Zn with preformed hydrozincite (HZ7) compared to the diamond polished surface (DP). In summary, without being able to consider the formation of simonkolleite, it can be concluded that an increased surface coverage of hydrozincite reduces the initial spreading ability of the NaCl-containing droplets and thereby lowers the overall formation rate of hydrozincite and cuprite.

B Example ScienceIE Training/Test Annotation Data

The following is ScienceIE test paper annotations file S0010938X15301268.ann:

T1 Material 46 64 corrosion products T2 Material 104 110 Cu40Zn
T3 Material 122 134 hydrozincite
T4 Material 149 153 Cu2O
T5 Material 378 384 Cu40Zn
T6 Material 408 410 DP
T7 Material 415 418 HZ7
T8 Material 530 536 Cu40Zn
T9 Material 552 564 hydrozincite
T10 Material 566 569 HZ7
* Synonym-of T9 T10
T11 Material 587 611 diamond polished surface
T12 Material 613 615 DP
* Synonym-of T11 T12
T13 Material 678 691 simonkolleite
T14 Material 751 763 hydrozincite
T15 Material 809 833 NaCl-containing droplets
T16 Material 883 895 hydrozincite
T17 Material 900 907 cuprite
T18 Process 456 471 formation rates
T20 Process 280 296 absorbance units
T19 Task 308 406 comparisons to be made of the amounts of each species between the two Cu40Zn surfaces investigated
R1 Hyponym-of Arg1:T3 Arg2:T1
R2 Hyponym-of Arg1:T4 Arg2:T1
T21 Material 480 492 hydrozincite
T22 Material 497 504 cuprite
T23 Process 665 691 formation of simonkolleite
T24 Process 776 793 initial spreading
T25 Process 865 879 formation rate