WINGNUS: Keyphrase Extraction Utilizing Document Logical Structure

Thuy Dung Nguyen
Department of Computer Science
School of Computing
National University of Singapore
nguyen14@comp.nus.edu.sg

Minh-Thang Luong
Department of Computer Science
School of Computing
National University of Singapore
luongmin@comp.nus.edu.sg

Abstract

We present a system description of the WINGNUS team work[1] for the SemEval-2010 task #5 Automatic Keyphrase Extraction from Scientific Articles. A key feature of our system is that it utilizes an inferred document logical structure in our candidate identification process, to limit the number of phrases in the candidate list, while maintaining its coverage of important phrases. Our top performing system achieves an F_1 of 25.22% for the combined keyphrases (author and reader assigned) in the final test data. We note that the method we report here is novel and orthogonal from other systems, so it can be combined with other techniques to potentially achieve higher performance.

1 Introduction

Keyphrases are noun phrases (NPs) that capture the primary topics of a document. While beneficial for applications such as summarization, clustering and indexing, only a minority of documents have manually-assigned keyphrases, as it is a time-consuming process. Automatic keyphrase generation is thus a focus for many researchers.

Most existing keyphrase extraction systems view this task as a supervised classification task in two stages: generating a list of candidates – *candidate identification*; and using answer keyphrases to distinguish true keyphrases – *candidate selection*. The selection model uses a set of features that capture the saliency of a phrase as a keyphrase. A major challenge of the keyphrase extraction task lies in the candidate identification process. A narrow candidate list will overlook some true keyphrases (favoring precision), whereas a broad list will produce more errors and require more processing in latter selection stage (favoring recall).

In our previous system (Nguyen and Kan, 2007), we made use of the document logical structure in the proposed features. The premise of this earlier work was that keyphrases are distributed non-uniformly in different logical sections of a paper, favoring sections such as *introduction*, and *related work*. We introduced features indicating which sections a candidate occurrs in. For our fielded system in this task (Kim et al., 2010), we further leverage the document logical structure for both candidate identification and selection stages.

Our contributions are as follows: 1) We suggest the use of Google Scholar-based crawler to automatically find PDF files to enhance logical structure extraction; 2) We provide a keyphrase distribution study with respect to different logical structures; and 3) From the study result, we propose a candidate identification approach that uses logical structures to effectively limit the number of candidates considered while ensuring good coverage.

2 Preprocessing

Although we have plain text for all test input, we posit that logical structure recovery is much more robust given the original richly-formatted document (*e.g.*, PDF), as font and formatting information can be used for detection. As a bridge between plain text data provided by the organizer and PDF input required to extract formatting features, we first describe our *Google Scholar-based crawler* to find PDFs given plain texts. We then detail on the *logical structure extraction* process.

Google Scholar-based Paper Crawler

Our crawler[2] takes inputs as titles to query Google Scholar (GS) by means of web scraping. It pro-

[1]This work was supported by a National Research Foundation grant "Interactive Media Search" (grant # R-252-000-325-279).

[2]http://wing.comp.nus.edu.sg/~lmthang/GS/

cesses GS results and performs approximate title matching using character-based *Longest Common Subsequence* similarity. Once a matching title with high similarity score (> 0.7 experimentally) is found, the crawler retrieves the list of available PDFs, and starts downloading until one is correctly stored. We enforce that PDFs accepted should have the OCR texts closely match the provided plain texts in terms of lines and tokens.

In the keyphrase task, we approximate the title inputs to our crawler by considering the first two lines of each plain text provided. For 140 train and 100 test input documents, the crawler downloaded 117 and 80 PDFs, of which 116 and 76 files are correct, respectively. This yields an acceptable level of performance in terms of (Precision, Recall) of (99.15%, 82.86%) for train and (95%, 76%) for test data.

Logical Structure Extraction

Logical structure is defined as "a hierarchy of logical components, for example, titles, authors, affiliations, abstracts, sections, etc." in (Mao et al., 2003). Operationalizing this definition, we employ an in-house software, called SectLabel (Luong et al., to appear), to obtain comprehensive logical structure information for each document. SectLabel classifies each text line in a scholarly document with a semantic class (*e.g.*, *title, header, bodyText*). Header lines are furthered classified into generic roles (*e.g.*, *abstract, intro, method*).

A prominent feature of SectLabel is that it is capable of utilizing rich information, such as font format and spatial layout, from an optical character recognition (OCR) output if PDF files are present[3]. In case PDFs are unavailable, SectLabel still handles plain text based logical structure discovery, but with degraded performance.

3 Candidate Phrase Identification

Phrase Distribution Study

We perform a study of keyphrase distribution on the training data over different logical structures (LSs) to understand the importance of each section within documents. These LSs include: *title, headers, abstract, introduction* (intro), *related work* (rw), *conclusion*, and *body text*[4] (body).

[3]We note that the PDFs have author assigned keyphrases of the document, but we filtered this information before passing to our keyphrases system to ensure a fair test.

[4]We utilize the comprehensive output of our logical structure system to filter out *copyright, email, equation, figure, caption, footnote*, and *reference* lines.

We make a key observation that within a paragraph, important phrases occur mostly in the first n sentences. To validate our hypothesis, we consider keyphrase distribution over $body_n$, which is the subset of all of the *body* LS, limited to the first n sentences of each paragraph ($n = 1, 2, 3$ experimentally).

	Ath	**Rder**	**Com**	**Sent**	**Den**
title	142	175	251	122	**2.06**
headers	158	342	425	1,893	0.22
abstract	276	745	897	1,124	**0.80**
intro	335	984	**1,166**	4,338	0.27
rw	160	345	443	1,945	0.23
concl	227	488	616	1,869	0.33
body	398	1,175	**1,411**	39,179	0.04
full	465	1,720	**1,994**	50,512	0.04
$body_1$	333	839	**1,035**	11,280	0.09
$body_2$	366	980	1,197	20,024	0.06
$body_3$	382	1,042	1,269	26,163	0.05
fulltext	480	1,773	**2,059**	166,471	0.01

Table 1: Keyphrase distribution over different logical structures computed from the 144 training documents. The type counts of author-assigned (ath), reader-assigned (rder) and combined (comb) keyphrases are shown. *Sent* indicates the number of sentences in each LS. The *Den* column gives the density of keyphrases for each LS.

Results in Table 1 show that individual LSs (title, headers, abstract, intro, rw, concl) contain a high concentration (i.e., density > 0.2) of keyphrases, with *title* and *abstract* having the highest density, and *intro* being the most dominant LS in terms of keyphrase count. With all these LSs and *body*, we obtain the *full* setting, covering 1994/2059=96.84% of all keyphrases appearing in the original text, *fulltext*, while effectively reducing the number of processed sentences by more than two-thirds.

Considering only the first sentence of each paragraph in the body text, $body_1$, yields fair keyphrase coverage of 1035/1411=73.35% relative to that of *fulltext*. The number of lines to be processed is much smaller, about a third, which validates our aforementioned hypothesis.

Keyphrase Extraction

Results from the keyphrase distribution study motivates us to further explore the use of logical structures (LS). The idea is to limit the search scope of our candidate identification system while maintaining coverage. We propose a new ap-

proach, which extracts candidates according to the regular expression rules discussed in (Kim and Kan, 2009). However, instead of using the whole document text as input, we abridge the input text at different levels from *full* to *minimal*.

Input	Description	Cand	Com	Recall
minimal	title + headers + abs + intro	30,702	1,312	63.72%
medium	*min* + rw + conclusion	44,975	1,414	68.67%
$full_1$	*med* + $body_1$	**73,958**	**1,580**	**76.74%**
$full_2$	*med* + $body_2$	90,624	1,635	79.41%
$full_3$	*med* + $body_3$	101,006	1,672	81.20%
full	*med* + body	121,378	1,737	84.36%
fulltext	original text	**148,411**	**1,766**	**85.77%**

Table 2: Different levels of abridged inputs computed on the training data. *Cand* shows the number of candidate keyphrases extracted for each input type; *Com* gives the number of correct keyphrases appear as candidates; *Recall* is computed with respect to the total number of keyphrases in the original texts (2059).

Results in Table 2 show that we could gather a recall of 63.72% when considering a significantly abridged form of the input culled from title, headers, abstract (abs) and introduction (intro) – *minimal*. Further adding related work (rw) and conclusion – *medium* – enhances the recall by 4.95%. When adding only the first line of each paragraph in the body text, we achieve a good recall of 76.74% while effectively reducing the number of candidate phrases to be process by a half with respect to the *fulltext* input. Even though $full_2$, $full_3$, and *full* show further improvements in terms of recall, we opt to use $full_1$ in our experimental runs, which trades off recall for less computational complexity, which may influence downstream classification.

4 Candidate Phrase Selection

Following (Nguyen and Kan, 2007), we use the Naïve Bayes model implemented in Weka (Hall et al., 2009) for candidate phrase selection. As different learning models have been discussed much previous work, we just list the different features with which we experimented with. Our features[5] are as follows (where *n* indicates a numeric feature; *b*, a boolean one):

[5]Detailed feature definitions are described in (Nguyen and Kan, 2007; Kim and Kan, 2009).

F1-F3 (*n*): TF×IDF, term frequency, term frequency of substrings.

F4-F5 (*n*): First and last occurrences (word offset).

F6 (*n*): Length of phrases in words.

F7 (*b*): Typeface attribute (available when PDF is present) – Indicates if any part of the candidate phrase has appeared in the document with bold or italic format, a good hint for its relevance as a keyphrase.

F8 (*b*): InTitle – shows whether a phrase is also part of the document title.

F9 (*n*): TitleOverlap – the number of times a phrase appears in the title of other scholarly documents (obtained from a dump of the DBLP database).

F10-F14 (*b*): Header, Abstract, Intro, RW, Concl – indicate whether a phrase appears in headers, abstract, introduction, related work or conclusion sections, respectively.

F15-F19 (*n*): HeaderF, AbstractF, IntroF, RWF, ConclF - indicate the frequency of a phrase in the headers, abstract, introduction, related work or conclusion sections, respectively.

5 Experiments

5.1 Datasets

For this task (Kim et al., 2010), we are given two datasets: *train* (144 docs) and *test* (100 docs) with detailed answers for *train*. To tune our system, we split the train dataset into train and validation subsets: $train_t$ (104 docs) and $train_v$ (40 docs). Once the best setting is derived from $train_t$-$train_v$, we obtain the final model trained on the full data, and apply it to the test set for the final results.

5.2 Evaluation

Our evaluation process is accomplished in two stages: we first experiment different feature combinations by using the input types *fulltext* and $full_1$. We then fix the best feature set, and vary our different abridged inputs to find the optimal one.

Feature Combination

To evaluate the performance of individual features, we define a *base* feature set, as $F_{1,4}$, and measure the performance of each feature added separately to the base. Results in Table 3 have highlighted the set of positive features, which is $F_{3,5,6,13,16}$.

From the positive set $F_{3,5,6,13,16}$, we tried different combinations for the two input types shown

System	F Score	System	F Score
base	*23.42%*	+ F_{11}	23.42%
+ F_2	21.13%	+ F_{12}	23.42%
+ F_3	**24.57%**	+ F_{13}	**23.75%**
+ F_5	**24.08%**	+ F_{14}	22.28%
+ F_6	**25.06%**	+ F_{15}	22.11%
+ F_7	23.42%	+ F_{16}	**23.59%**
+ F_8	22.77%	+ F_{17}	22.60%
+ F_9	22.28%	+ F_{18}	23.26%
+ F_{10}	23.42%	+ F_{19}	21.95%

Table 3: Performance of individual features (on *fulltext*) added separately to the base set $F_{1,4}$.

in Table 4. The results indicate that while *fulltext* obtains the best performance with $F_{3,6,5}$ added, using *full*$_1$ shows superior performance at 28.18% F Score with $F_{3,6}$ added. Hence, we have identified our best feature set as $F_{1,3,4,6}$.

	fulltext	**full**$_1$
base ($F_{1,4}$)	23.42%	22.60%
+ $F_{3,6}$	25.88%	**28.18%**
+ $F_{3,6,5}$	**26.21%**	26.21%
+ $F_{3,6,5,13}$	24.90%	26.21%
+ $F_{3,6,5,16}$	24.24%	26.70%
+ $F_{3,6,5,13,16}$	23.42%	26.70%

Table 4: Performance (F_1) over difference feature combinations for *fulltext* and *full*$_1$ inputs.

Abridged Inputs

Table 5 gives the performance for the abridged inputs we tried with the best feature set $F_{1,3,4,6}$. All *full*$_1$, *full*$_2$, *full*$_3$ and *full* show improved performance compared to those on the *fulltext*. We achieve our best performance with *full*$_1$ at 28.18% F Score. These results validate the effectiveness of our approach in utilizing logical structure for the candidate identification. We report our results submitted in Table 6. These figures are achieved using the best feature combination $F_{1,3,4,6}$.

6 Conclusion

We have described and evaluated our keyphrase extraction system for the SemEval-2 Task #5. With the use of logical structure in the candidate identification, our system has demonstrated its superior performance over systems that do not use such information. Moreover, we have effectively reduced the numbers of text lines and candidate phrases to be processed in the candidate identification and selection respectively by about half.

Input	**@5**	**@10**	**@15**	**Fscore**
min	62	110	145	23.75%
med	79	130	158	25.88%
full$_1$	84	135	172	**28.18%**
full$_2$	90	132	164	26.86%
full$_3$	89	134	162	26.54%
full	84	130	164	26.86%
fulltext	82	127	158	**25.88%**

Table 5: Performance over different abridged inputs using the best feature set $F_{1,3,4,6}$. "@N" indicates the number of top N keyphrase matches.

System	**Description**	**F@5**	**F@10**	**F@15**
WINGNUS$_1$	full, $F_{1,3,4,6}$	20.65%	24.66%	24.95%
WINGNUS$_2$	full$_1$, $F_{1,3,4,6}$	20.45%	24.73%	**25.22%**

Table 6: Final results on the test data.

Our system takes advantage of the logical structure analysis but not to the extent we had hoped. We had hypothesized that formatting features (F_7) such as bold and italics, would help discriminate key phrases, but in our limited experiments for this task did not validate this. Similarly, external knowledge should help in the keyphrase task, but the prior knowledge about keyphrase likelihood (F_9) in DBLP hurt performance in our tests. We plan to further explore these issues for the future.

References

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Su Nam Kim and Min-Yen Kan. 2009. Re-examining automatic keyphrase extraction approaches in scientific articles. In *MWE '09*.

Su Nam Kim, Alyona Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Task 5: Automatic keyphrase extraction from scientific articles. In *SemEval*.

Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. to appear. Logical structure recovery in scholarly articles with rich document features. IJDLS. Forthcoming, accepted for publication.

Song Mao, Azriel Rosenfeld, and Tapas Kanungo. 2003. Document structure analysis algorithms: a literature survey. In *Proc. SPIE Electronic Imaging*.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *ICADL*.