SIG Proceedings Paper in LaTeX Format*

Extended Abstract†

ABSTRACT

The tremendous amount of user-generated contents, such as the questions about symptoms and diseases being asked by a user in online forums, or blog posts about dietary and fitness, are potential sources for enriching knowledge in medical document retrieval and health-related text mining applications. Understanding those various contents is simply not a trivial task. Reducing the complex characteristics of contents, including but not limited to linguistic variations, into structured information can help the text interpretation process. A common approach is to extract medical terms from each document using available tools, such as medical entity recognizer. However, we argue that there are many missing pieces of information when we rely only on medical entities or keywords as main information. Hence, we use keyphrases extraction techniques to extract more important information about the concept of the document. Previous works in keyphrases extraction mostly focused on a general domain, long and formal written documents. These methods perform poorly when being applied to user-generated contents and medical documents. In this work, we propose deep bi-directional recurrent network and convolutional neural network as a model for extracting keyphrases. Furthermore, we leveraged word embedding, medical concepts from a knowledge base, and linguistic components as our features. The proposed algorithm achieved F1 score of 82.41%. This result outperforms state of the art system for extracting keyphrases using a statistical model.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference format:

. 1997. SIG Proceedings Paper in LaTeX Format. In *Proceedings of ACM Woodstock conference, El Paso, Texas USA, July 1997 (WOODSTOCK'97),* 7 pages.

https://doi.org/10.475/123_4

*Produces the permission block, and copyright information

†The full version of the author's guide is available as `acmart.pdf` document

WOODSTOCK'97, July 1997, El Paso, Texas USA

ACM ISBN 123-4567-24-567/08/06…$15.00

https://doi.org/10.475/123_4

1 INTRODUCTION

There are tremendous amount of user-generated contents about medical information through the internet. As a result, there is a growing need for enriching medical-related knowledge, for which it can be very valuable for several tasks, such as medical document retrieval, medical question answering, and health-related text mining applications. However, understanding those various health-related contents, especially from social media, is simply not a trivial task since they are written in free text format (i.e. unstructured format), and often prone to grammatical and typographical glitches. Therefore, reducing the complex characteristics of the textual contents, including but not limited to linguistic variations, into structured information could be very beneficial for the text interpretation process.

One possible solution is to extract medical terms that represents the document they belong to. The common approach is harnessing medical entity recognizer. However, we argue that there are many missing pieces of information when we merely rely on medical entities or terminologies as our main information for representing a particular document. In this paper, we present new keyphrases extraction techniques to extract more important information about the concept expressed inside the document. Furthermore, we also propose a new definition of keyphrases, which is based on our observation of health-related documents. Actually, our final goal is to develop medical question answering systems to assist physicians or people obtaining health information. The work presented in this paper is absolutely inline with our goal, since important keyphrases of a question document can be used to formulate a query for the passage retrieval subsystem.

Keyphrases are usually selected phrases or clauses that can capture the main topic of a given document [17]. They can provide readers with highly valuable and representative information, such that looking at keyphrases is sufficient to understand the whole body of document. Moreover, keyphrases are important for text summarization, document clustering, and classification task ([4]; [11]; [3]). Previous works have shown that using keyphrases to generate queries can improve the quality of question retrieval in medical question answering forum [1].

Keyphrase extraction methods have been mostly using statistical approach ([16] [21] [13] [10]) and supervised learning approach ([18] [8] [7]). The statistical approach often involves keyword candidate selection and candidate's property calculation. The "keywordness" value of a candidate phrase is usually scored based on its typical length as well as frequency in which document it appears [13]. While the statistical approach is very efficient to generate keyphrases, our experiments showed that it performs poorly on user generated contents and short documents. The other approach is based on supervised learning, there are two common approach in supervised learning: classification and sequence labeling. In classification approach, candidate keyphrases are extracted using statistical [16] or language model [15]. Then, the candidate keyphrases

are classified whether a candidate is a keyphrase or not by using a pretrained model. However, classification approach cannot capture semantic relationship between words in a document [9]. In sequence labeling approach, like the one proposed by [1] and [19]. They define keyphrase extraction task as a sequence labeling task, in which each word in the document can have two possible labels: keyword or non-keyword. The method they proposed worked well by outperforming unsupervised approach. However, synonym and ambiguity words cannot be handled properly [19].

In our work, we use the same approach as [1] which treats keyphrase extraction as a sequence labeling task. In our experiment, we employed and combined various deep learning architectures, such as convolutional neural networks and bi-directional long short-term memory networks, to exploit high level features between neighboring word positions. To improve the quality of our model, we leverage several new hand-crafted features that can handle our keyphrase extraction problems in medical user-generated content, such as word importance and word stickiness features. The word importance feature is enable to rank words in a document by their importance value. The more important a word is to the content of document, the higher its "keywordness" value would seem to be. In addition, we also propose word stickiness feature to make our model constructs keyphrases better.

We also employed pre-trained word embedding to incorporate contextual, semantic, and syntactic information of a word. For example, we found that pre-trained word embedding can handle slang words and abbreviations by giving their word vectors closer to the vectors of their canonical forms. Finally, to evaluate the effectiveness of our model, we collected and manually annotated data from Indonesian medical forum questions, such as alodokter, health detik, doktergratis, dokter.id, doktersehat, klikdokter [1]. After that, the dataset was used for both training and testing our computational models. For our baseline, we use RAKE [13], which is the current state of the art model for extracting keyphrases.

This paper is organized as follows. In section 2 we discuss related works in keyphrase extraction. In section 3, the proposed keyphrase extraction method has been discussed. We present the evaluation and the experimental results in section 4.

2 RELATED WORKS

In general, keyphrase extraction methods can be divided into two groups: unsupervised ranking (statistical) approach and supervised machine learning approach. For the unsupervised line of research, keyphrase extraction is formulated as a ranking problem, in which each candidate keyphrase is assigned a score that represents its keywordness value. Furthermore, this kind of approach does not require training data, which is often very difficult to obtain. On the other hand, supervised machine learning approach requires training data that contains a collection of documents with their labeled keywords. Using this approach, keyphrase extraction is usually treated as a classification or sequence labeling task in the level of words or phrases. The first step of this approach generates candidate keyphrases from a particular document. Finally, every candidate keyphrase in the document will be classified as either a keyphrase or non-keyphrase. Recently, a well-known supervised approach for keyphrase extraction is based on sequence labeling problem ([19][1][20]). The assumption behind this model is that the decision on whether a particular word serves as a keyword is affected by the information from its neighboring word positions.

In the work of [16], they leverage TF-IDF to get the most significant words. Specifically, they utilize term frequency and the inverse of document frequency to rank terms in the document. First, this Rapid Automatic Keywords Extraction [13] uses stopwords to split a document into several candidate keyphrases. All candidates will be ranked by the degrees and frequencies of all words inside them. Finally, the rank points of each candidate is obtained by summing over all degrees and frequencies of all words contained in each keyphrase. RAKE is proven to be successful for extracting keyphrases in special documents such as paper and article [13].

There is also a popular graph-based approach, namely TextRank (U), i.e., a modified PageRank algorithm to extract keyphrases from a document. They exploit word co-occurrence information to build a document graph, in which a word serves as a node and co-occurence information determines an edge between two nodes (words). After they build the graph, they actually run PageRank algorithm to determine the keywordness score for each node in the graph. A word with high PageRank value denotes that its related concept can be found in many location inside the document, which means that it is a good candidate for keyword. After that, keyphrases can be obtained using post-processing step that may combine contiguous candidates into one keyphrase.

One of the earlier supervised approaches in the problem of keyphrase extraction is KEA [18]. KEA uses tf-idf and first occurrence of the term to identify candidate keyphrases. They use a machine-learning algorithm (i.e., Naive Bayes) to predict whether or not a candidate is a good keyphrase. [8] developed a system called MAUI that can extract keyphrases using trained corpus and controlled vocabulary. MAUI itself is an improvement to the aforementioned supervised model, KEA. Furthermore, [7] use word embedding and brown clustering as features. Their experiment result had shown that the proposed features and MAUI performs better than statistical methods, such as simple tf-idf approach.

The utilization of neural networks is also quite popular in the recent years. [15] uses Feed Forward Neural Networks to classify candidate keyphrases. To extract candidate keyphrases, they use POS tag information for classifying all words in the document. Then, Deterministic Finite Automation (DFA) is used to extract noun phrases as candidate keyphrases. Furthermore, deep neural networks is also employed for extracting keyphrase, like the one proposed by [20], that uses joint Recurrent Neural Networks (RNNs) layer to capture keyphrase sequences in microblogs. The research had shown that RNN is very effective to extract keyword.

Unfortunately, there are limited works regarding the task of keyphrase extraction in medical domain. [14] use a hybrid medical knowledge base and statistical approach to extract keyphrases from medical articles. They use stopwords to split candidate keyphrases, then candidates are ranked with two aspects: PF-IDF (Phrase Frequency * Inverse Document Frequency) and domain knowledge which is extracted from medical article. In terms of medical social

[1] www.alodokter.com, health.detik.com, www.dokter.id, www.doktersehat.com, www.klikdokter.com

Table 1: Statistical information of dataset. W, K, $\bar{N}_w$, $\bar{N}_k$ are the total of words, number of keyphrases, average number of words, and average number of keyphrases in each question, respectively.

questions	W	K	$\bar{N}_w$	$\bar{N}_k$
416	26747	64.76	1861	4.49

media content, [1] use Conditional Random Fields (CRFs) for extracting keywords from medical questions in online health-forum. They use information, such as word location and length as features in their experiments.

3 METHODOLOGY

3.1 DATA AND ANNOTATION

To analyze the effectiveness of our model for keyphrase extraction in user-generated medical domain, we constructed an evaluation dataset. A a number of questions was gathered from Indonesian medical question answering forum. The data was crawled by [5] and [12] in their experiment. Generally, we gathered 416 question-answer pairs from [5] and [12]. We decided to use the same question-answers pairs to observe our keyphrases and medical entities that has been annotated by them. In addition, system for recognizing medical entities is not fully build, by using the same document we can use the annotated medical entities as our feature.

From analyzing these question-answer pairs, we found some noise and unnecessary information. Question and answer have different main topic and focus. Moreover, we are focusing our task for getting user intentions and needs in our experiment. Based on that problem, we only use questions by users and remove answers by the doctors because it would be a better representation of user intentions and needs. There is also unnecessary information such as URL and email. We replace URL with âĂIJURLâĂİ and email with âĂIJemailâĂİ since we were focusing on textual content. Moreover, we also removed ads and spam in the data we gathered.

Because we want to process all the questions by users, we tokenize every question in our data. The tokenization process included separating alphanumeric with non-alphanumeric. Then separating alphanumeric with numeric.

To evaluate the quality of our model. We manually annotated or data to label keyphrases in 416 questions. In this work, we use "IOB" tagging scheme to label every word [2]. In Table X, we give an example sequence with labels for each word. The statistical information of the dataset can be seen in Table X

3.2 TASK DEFINITIONS

3.3 MODEL ARCHITECTURE

We process the sequences word by word. Features are extracted from every word as the input representation of the model. The output for this words are label which describe a word is either part of keyphrase or not.

To give our model a better insight of context, we introduce Convolutional Neural Networks (CNNs) to capture context from three adjacent words. From our observation, to know whether a word is part of keyphrase or not, we need to know the context of the word itself. For example, âĂIJDoc, i have a frequent back pain. What happen?âĂİ. The keyphrase for previous example is âĂIJfrequent back painâĂİ. In order to know that âĂIJbackâĂİ is part of keyphrase, we need information from the word âĂIJfrequentâĂİ which give an intensity of a symptom and âĂIJpainâĂİ which refer to âĂIJback painâĂİ as main term. With CNNs, we argue that our model can capture the surrounding information of a word.

To get a better inference, information from the past and future sequences can be integrated. This approach is proven effective in sequence labeling task such as semantic role labeling [22] and named entity recognition [6]. Hence, we utilized bi-directional LSTM (B-LSTM) for extracting structural knowledge by processing sequences both forward and backward. To perform the sequence tagging task, we build a fully connected layer in the top of our model. This layer will classify the output of our model with softmax activation function.

We also experimented on customizing the way we input the features underneath our aforementioned networks. Instead of concatenating all feature into one vector, we tried to give weight for every feature we have. We argue that each feature have different contribution to the model. In order to do so, we create a new layer underneath our model to do the weighting scenario. The following equation to weight all feature can be seen in below:

$$Z = tanh(W_1 * F_1 + W_2 * F_2 + .. + W_n * F_n) \quad (1)$$

Where W_i are the weight for each feature and F_i are the extracted feature in vector form.

4 EXPERIMENTS

In this section, we analyzed the importance of our features by conducting a feature ablation study. Furthermore, we also established a scenario to test the performance of our model.

4.1 Feature Ablation Study

In this part, perform a feature ablation study. By considering all the features, and then removing one of these features, we want to estimate the importance of the features. To the experiment. We use 80% as training set, and 20% as a testing set. We used precision (P), recall (R), and F1-score (F1) as evaluation metrics. Table 2 show the experimental result. Then, we use LSTM as evaluation model because it is the most simple architecture. These are the following features we proposed: word stickiness (Ws), word abbreviation and acronym (Wa), Medical Entities (Me), Word Location (Wp), Word Length (Wl), Word Importance (Wi), POS tag (Pt), Medical Dictionary (Md).

The experimental results in Table 2 shows that every feature has a positive contribution to the model. It is shown by iteratively remove each feature one by one and every time we remove a feature, F-1 score always dropped. Hence, we use every feature to train our model in the next experiments.

4.2 Model Scenario

We created a scenario to evaluate the performance of our model. In this scenario, we also compared our model to an unsupervised learning algorithm using RAKE. Moreover, we also implemented CRF

Table 2: Feature Ablation Study Result

Features	Precision	Recall	F-Measure
We + Ws + Wa + Me + Wp + Wl + Wi + Pt + Md	79.93%	57.45%	66.85%
Ws + Wa + Me + Wp + Wl + Wi + Pt + Md	84.01%	75.79%	79.69%
We + Wa + Me + Wp + Wl + Wi + Pt + Md	76.39%	73.59%	74.96%
We + Ws + Me + Wp + Wl + Wi + Pt + Md	78.44%	73.83%	76.07%
We + Ws + Wa + Wp + Wl + Wi + Pt + Md	79.38%	76.28%	77.8%
We + Ws + Wa + Me + Wl + Wi + Pt + Md	74.24%	78.23%	76.19%
We + Ws + Wa + Me + Wp + Wi + Pt + Md	75.36%	76.28%	75.82%
We + Ws + Wa + Me + Wp + Wl + Pt + Md	83.69%	75.3%	79.27%
We + Ws + Wa + Me + Wp + Wl + Wi + Md	79.68%	74.81%	77.17%
We + Ws + Wa + Me + Wp + Wl + Wi + Pt	75.66%	76.03%	75.85%

Table 3: Model Scenario Result

Models	Precision	Recall	F-Measure
RAKE	43.24%	68.19%	52.90%
CRF	63.44%	64.94%	62.52%
LSTM	78.77%	79.71%	79.16%
B-LSTM	81.88%	83.05%	82.37%
CNN-B-LSTM	82.00%	82.99%	82.41%

to as a baseline for this sequence labeling task. CRF also has been proven effective in extracting keyphrases [1] [19]. The following scenarios are listed below:

- RAKE
- CRF
- B-LSTM
- CNN-B-LSTM
- Weighting-CNN-B-LSTM

The result of the scenario can be seen in Table 3.

5 INI CONTOH

5.1 Type Changes and *Special* Characters

We have already seen several typeface changes in this sample. You can indicate italicized words or phrases in your text with the command `\textit`; emboldening with the command `\textbf` and typewriter-style (for instance, for computer code) with `\texttt`. But remember, you do not have to indicate typestyle changes when such changes are part of the *structural* elements of your article; for instance, the heading of this subsection will be in a sans serif[2] typeface, but that is handled by the document class file. Take care with the use of[3] the curly braces in typeface changes; they mark the beginning and end of the text that is to be in the different typeface.

You can use whatever symbols, accented characters, or non-English characters you need anywhere in your document; you can find a complete list of what is available in the *LaTeX User's Guide* [**?**].

[2]Another footnote here. Let's make this a rather long one to see how it looks.
[3]Another footnote.

5.2 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

5.2.1 Inline (In-text) Equations. A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin . . . \end` construction or with the short form `$ . . . $`. You can use any of the symbols and structures, from α to ω, available in LaTeX [**?**]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\to\infty} x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

5.2.2 Display Equations. A numbered display equation—one set off by vertical space from the text and centered horizontally—is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\to\infty} x = 0 \tag{2}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty} x_i = \int_0^{\pi+2} f \tag{3}$$

just to demonstrate LaTeX's able handling of numbering.

5.3 Citations

Citations to articles [**? ? ? ?**], conference proceedings [**?**] or maybe books [**? ?**] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the

Table 4: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

proper location in the `.tex` file [**?**]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide* by Lamport [**?**].

This article shows only the plainest form of the citation command, using `\cite`.

5.4 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material are found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 4 is included in the input file; compare the placement of the table here with the table in the printed output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 5 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed output of this document.

It is strongly recommended to use the package booktabs [**?**] and follow its main principles of typography with respect to tables:

(1) Never, ever use vertical rules.
(2) Never use double rules.

It is also a good idea not to overuse horizontal rules.

5.5 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of `.eps` files to be displayable with LaTeX. If you work with pdfLaTeX, use files in the `.pdf` format. Note that most modern TeX systems will convert `.eps` to `.pdf` for you on the fly. More details on each of these are found in the *Author's Guide*.

Figure 1: A sample black and white graphic.

Figure 2: A sample black and white graphic that has been resized with the `includegraphics` command.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure*** to enclose the figure and its caption. And don't forget to end the environment with **figure***, not **figure**!

5.6 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. ACM uses two types of these constructs: theorem-like and definition-like.

Here is a theorem:

Theorem 5.1. *Let f be continuous on $[a, b]$. If G is an antiderivative for f on $[a, b]$, then*

$$\int_a^b f(t)\,dt = G(b) - G(a).$$

Here is a definition:

Definition 5.2. If z is irrational, then by e^z we mean the unique number that has logarithm z:

$$\log e^z = z.$$

The pre-defined theorem-like constructs are **theorem**, **conjecture**, **proposition**, **lemma** and **corollary**. The pre-defined definition-like constructs are **example** and **definition**. You can add your own constructs using the *amsthm* interface [**?**]. The styles used in the `\theoremstyle` command are **acmplain** and **acmdefinition**.

Another construct is **proof**, for example,

Proof. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c} \left[gx \cdot \frac{f(x)}{g(x)}\right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

Table 5: Some Typical Commands

Command	A Number	Comments
`\author`	100	Author
`\table`	300	For tables
`\table*`	400	For wider tables

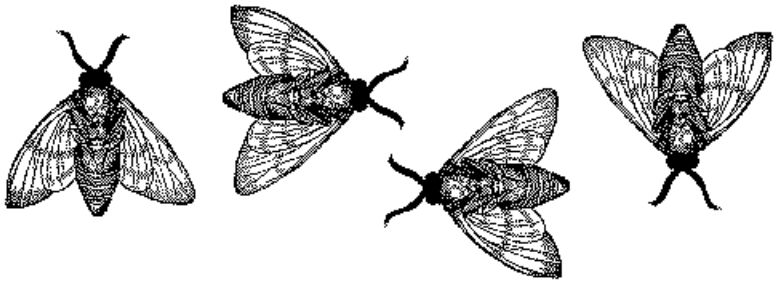

Figure 3: A sample black and white graphic that needs to span two columns of text.

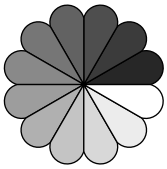

Figure 4: A sample black and white graphic that has been resized with the `includegraphics` command.

6 CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

A HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e., the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters.

A.2.2 Math Equations.

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations.

A.2.4 Tables.

A.2.5 Figures.

A.2.6 Theorem-like Constructs.

A Caveat for the TEX Expert.

A.3 Conclusions

A.4 References

Generated by bibtex from your `.bib` file. Run latex, then bibtex, then latex twice (to resolve references) to create the `.bbl` file. Insert that `.bbl` file into the `.tex` source file and comment out the command `\thebibliography`.

B MORE HELP FOR THE HARDY

Of course, reading the source code is always useful. The file `acmart.pdf` contains both the user guide and the commented code.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Yuhua Li for providing the matlab code of the *BEPS* method.

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is supported by the National Natural Science Foundation of China under Grant No.: 61273304 and Young Scientsts' Support Program (http://www.nnsf.cn/youngscientsts).

REFERENCES

[1] Yong-gang Cao, James J Cimino, John Ely, and Hong Yu. 2010. Automatically extracting information needs from complex clinical questions. *Journal of biomedical informatics* 43, 6 (2010), 962–971.

[2] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[3] Zhiguo Gong and Qian Liu. 2009. Improving keyword based web image search with visual feature distribution and term expansion. *Knowledge and Information Systems* 21, 1 (2009), 113–132.

[4] Khaled M Hammouda, Diego N Matute, and Mohamed S Kamel. 2005. Corephrase: Keyphrase extraction for document clustering. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. Springer, 265–274.

[5] Raditya Herwando. 2016. Pengenalan Entitas Kesehatan pada Forum Kesehatan Online Berbahasa Indonesia Menggunakan Algoritma Conditional Random Fields. (7 2016).

[6] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).

[7] Luis Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W Black, Anatole Gershman, David Martins de Matos, João Paulo da Silva Neto, and Jaime G Carbonell. 2015. Automatic Keyword Extraction on Twitter.. In *ACL (2)*. 637–643.

[8] Olena Medelyan. 2009. *Human-competitive automatic topic indexing*. Ph.D. Dissertation. The University of Waikato.

[9] Zakariae Alami Merrouni, Bouchra Frikh, and Brahim Ouhbi. 2016. Automatic keyphrase extraction: An overview of the state of the art. In *Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on*. IEEE, 306–313.

[10] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. Association for Computational Linguistics.

[11] Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 563–570.

[12] Wahid Nur Rohman. 2017. Pengenalan Entitas Kesehatan pada Forum Kesehatan Online dengan Menggunakan Recurrent Neural Networks. (1 2017).

[13] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining* (2010), 1–20.

[14] Kamal Sarkar. 2013. A hybrid approach to extract keyphrases from medical documents. *arXiv preprint arXiv:1303.1441* (2013).

[15] Kamal Sarkar, Mita Nasipuri, and Suranjan Ghose. 2010. A new approach to keyphrase extraction using neural networks. *arXiv preprint arXiv:1004.3274* (2010).

[16] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[17] Peter D Turney. 2000. Learning algorithms for keyphrase extraction. *Information retrieval* 2, 4 (2000), 303–336.

[18] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 1999. KEA: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*. ACM, 254–255.

[19] Chengzhi Zhang. 2008. Automatic keyword extraction from documents using conditional random fields. *Journal of Computational Information Systems* 4, 3 (2008), 1169–1180.

[20] Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on Twitter. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 836–845.

[21] Yongzheng Zhang, Evangelos Milios, and Nur Zincir-Heywood. 2007. A comparative study on key phrase extraction methods in automatic web site summarization. *Journal of Digital Information Management* 5, 5 (2007), 323.

[22] Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks.. In *ACL (1)*. 1127–1137.