Detection and Analysis of Ambiguities in Software Requirement Specification

Prerana Chaithra[1], Dr. Shantharam Nayak[2]

[1]VTU Research Scholar, RVCE & Associate Professor, Information Science & Engineering, Sapthagiri College of Engineering, affiliated to VTU, Bengaluru
[2]Professor & Principal advisor, affiliated to Visveswaraya Technological University, Bengaluru
[1]preranachaithra15@gmail.com, [2]shantaram_nayak@yahoo.com

Abstract:

Software requirement specification (SRS) forms an important document for capturing the details of newly designed software. Since natural language is used for representing, this enables the end-user to assess the suitability of the software for the application. Hence, the quality of SRS document is of great importance and evaluation of the document plays a significant role. The present study aims at identifying the lexical and syntactic ambiguities in the SRS document. Identification of ambiguities is carried out by tagging the extracted words from the document assisted by natural language processing. Python with in-built Natural Language Tool Kit (NLTK) is used for extracting the parts-of-speech (POS), while storing and retrieving the data is carried out with the aid of MongoDB database. The ambiguity levels are defined using the Ambiguity datahandbook[1] and the document ambiguity is assessed. The overall ambiguity percentage is specified and the changes necessary for the improvement of the document are indicated.

***Keywords:* Software Requirement Specification(SRS), Lexical Ambiguity(LE), Syntactic Ambiguity(SE), Natural Language Processing (NLP), Parts-of-speech (POS), stop-words, tokenization**

1. Introduction

Since the advent of computers, several softwares have been designed and developed to address different applications. Softwares are generic in nature which can be extended to any field of application viz. engineering, medical, commerce and so on. The present generation computers are extremely complex and pave way for more complicated softwares. There are numerous softwares available, which demands lucid documentation, free from ambiguities. Documentation of the strengths and weaknesses of software is as significant as designing software.

Software requirement specification is a document which exhibits the details and specification pertaining to software. Majority of documents are written in natural language[2], which introduces imprecision in the document. Therefore, assessment of the quality of SRS document plays an important role, which has gained importance over a decade. There are significant efforts put-forth to improve the quality of the SRS document [3][4][5][6]1. The present study explains a tool developed to identify and explain the ambiguity encountered in an SRS document. The approach renders a small step towards

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

satisfying the requirements of an ambiguity detection tool[7].

2. Literature Survey

There have been numerous attempts to identify and analyze the ambiguities encountered in a document. Some the notable efforts worth indicating involves development of tools for different applications pertaining to various domains[8][9][10][11]. Also, over a decade and half, significant attempts have been made to identify the analyze the ambiguities in requirement specifications to facilitate their automatic detection [12][13][14][16][17]. However, the attempts can be further extended to obtain reliable detection tools for SRS documents, assisting in easy detection of ambiguities and improvement of the document quality.

The present effort aims at developing a tool which not only identifies the ambiguity, but also explains the lines on which it can be improved. The tool indicates the line number where the ambiguity exists and the level of ambiguity listed as per the ambiguity datahandbook[1]. It also identifies the synonyms for the nouns and verbs, facilitated by parts-of-speech (POS) tagging, which assists in finding the right word to eliminate ambiguity in the sentence. After each cycle of processing, the percentage of ambiguity is calculated thereby quantifying the ambiguity existing in the document. This also enables to concentrate on those areas in the document where the ambiguity has serious consequences on the quality of the document. It also saves time taken for eliminating the ambiguity by identifying the right sentences or paragraphs.

3. Methodology

The proposed model is based on machine learning [15]. The machine learning algorithm takes SRS document as input and provides output with ambiguous words highlighted. The model training has been done using a huge database of SRS documents. The model is trained and tested for accuracy. Figure 1 indicates the flow of data in the proposed model.

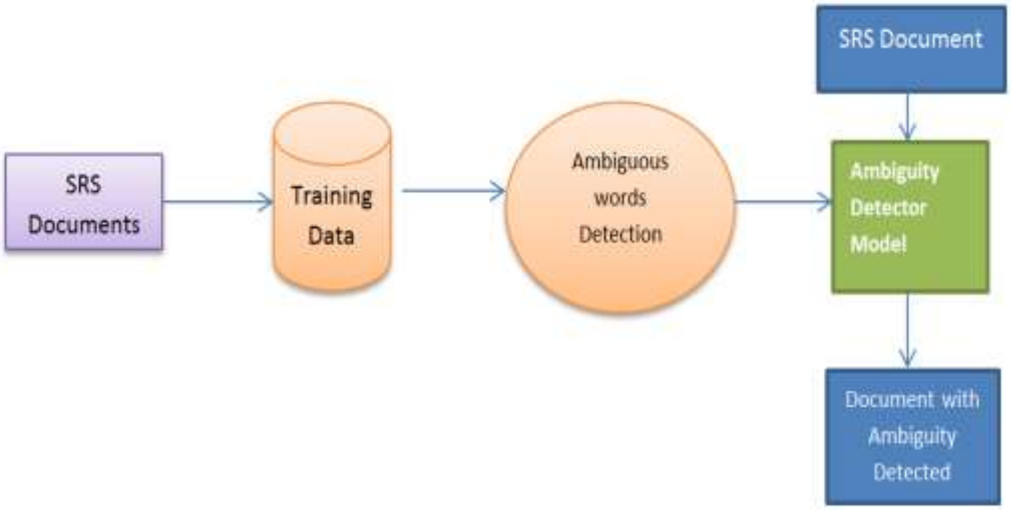


Figure 1: The Proposed Model

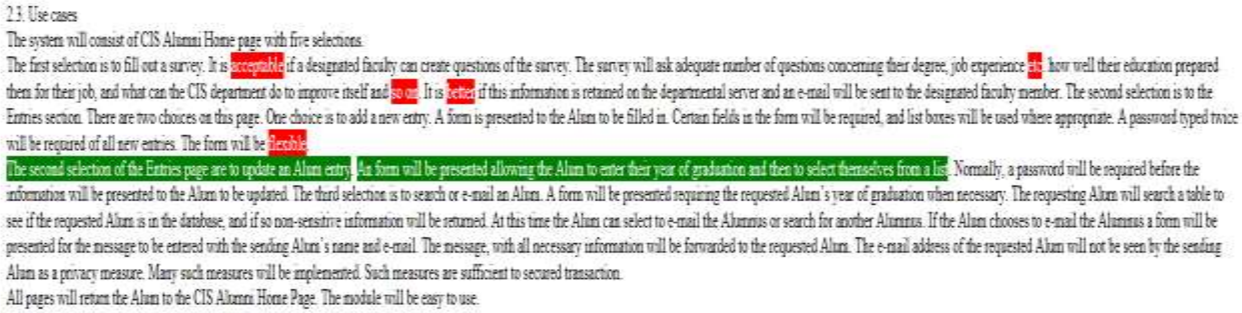
Algorithm is as follows

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

Step 1	**Start** Input: Software requirement specification documents.
Step 2	**Identification of ambiguous words** Ambiguous words to be detected in the document are identified.
Step 3	**Model building using training data** The documents are divided into training data and testing data. 70% of data is used as training data and it is fed to the model.
Step 4	**Testing the model accuracy** Testing data (30% of data) (SRS documents) are fed to the model.
Step 5	**Input SRS documents** The real data (SRS documents) are fed to the pre developed model.
Step 6	**Detection of ambiguous words** SRS specific and general ambiguity in the document are highlighted.
Step 7	**Analysis of ambiguous words** Analysis of lexical and semantic ambiguities is done.
Step 8	**Indicate the necessary changes for the improvement of the document** Suggest the changes required for modifying the SRS document to reduce ambiguity.
Step 9	**End**

4. Results

The results depict that the ambiguous words are identified. After the identification of the ambiguous words, they are analyzed to which category they belong to. As indicated in the previous section, SRS document is parsed and the sentences are stored in the database in JSON format.



2.3. Use cases

The system will consist of CIS Alumni Home page with five selections.

The first selection is to fill out a survey. It is acceptable if a designated faculty can create questions of the survey. The survey will ask adequate number of questions concerning their degree, job experience etc. how well their education prepared them for their job, and what can the CIS department do to improve itself and so on. It is better if this information is retained on the departmental server and an e-mail will be sent to the designated faculty member. The second selection is to the Entries section. There are two choices on this page. One choice is to add a new entry. A form is presented to the Alum to be filled in. Certain fields in the form will be required, and list boxes will be used where appropriate. A password typed twice will be required of all new entries. The form will be flexible.

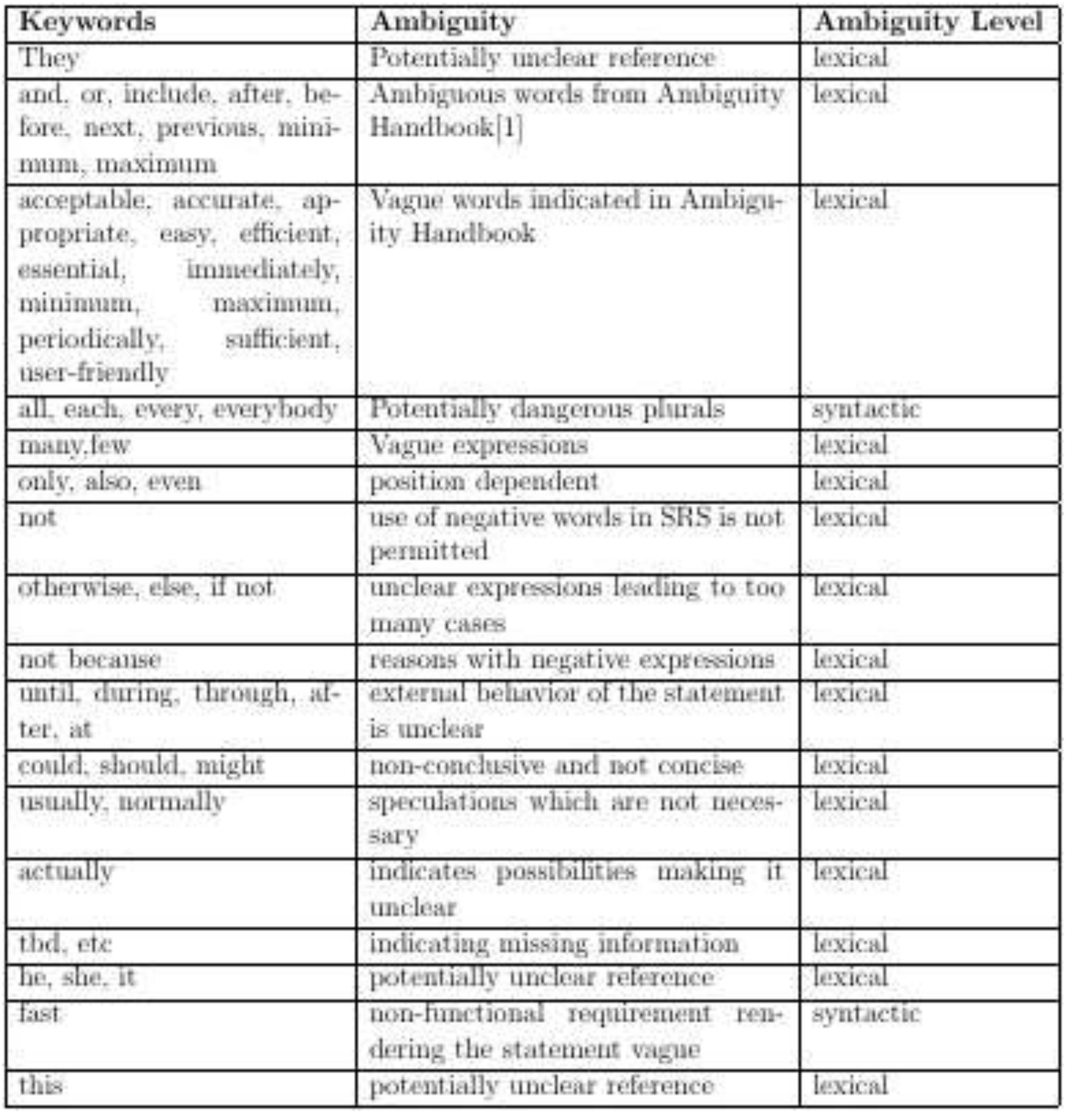
The second selection of the Entries page are to update an Alum entry. An form will be presented allowing the Alum to enter their year of graduation and then to select themselves from a list. Normally, a password will be required before the information will be presented to the Alum to be updated. The third selection is to search or e-mail an Alum. A form will be presented requiring the requested Alum's year of graduation when necessary. The requesting Alum will search a table to see if the requested Alum is in the database, and if so non-sensitive information will be returned. At this time the Alum can select to e-mail the Alumnus or search for another Alumnus. If the Alum chooses to e-mail the Alumnus a form will be presented for the message to be entered with the sending Alum's name and e-mail. The message, with all necessary information will be forwarded to the requested Alum. The e-mail address of the requested Alum will not be seen by the sending Alum as a privacy measure. Many such measures will be implemented. Such measures are sufficient to secured transaction.

All pages will return the Alum to the CIS Alumni Home Page. The module will be easy to use.

Figure 3: Detecting words which lead to ambiguity

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

Table – 1: Keywords for identification of ambiguities [5]



Keywords	Ambiguity	Ambiguity Level
They	Potentially unclear reference	lexical
and, or, include, after, before, next, previous, minimum, maximum	Ambiguous words from Ambiguity Handbook[1]	lexical
acceptable, accurate, appropriate, easy, efficient, essential, immediately, minimum, maximum, periodically, sufficient, user-friendly	Vague words indicated in Ambiguity Handbook	lexical
all, each, every, everybody	Potentially dangerous plurals	syntactic
many,few	Vague expressions	lexical
only, also, even	position dependent	lexical
not	use of negative words in SRS is not permitted	lexical
otherwise, else, if not	unclear expressions leading to too many cases	lexical
not because	reasons with negative expressions	lexical
until, during, through, after, at	external behavior of the statement is unclear	lexical
could, should, might	non-conclusive and not concise	lexical
usually, normally	speculations which are not necessary	lexical
actually	indicates possibilities making it unclear	lexical
tbd, etc	indicating missing information	lexical
he, she, it	potentially unclear reference	lexical
fast	non-functional requirement rendering the statement vague	syntactic
this	potentially unclear reference	lexical

Figure 3 depicts the words which lead to ambiguity based on keywords in the table 1. This facilitates easy detection and improvement of the SRS document by replacing the indicated words with the synonyms demonstrated above.

5. Conclusion

Tool for detecting and analyzing syntactic and lexical ambiguities is developed. Though it is difficult to nullify all the ambiguities in the SR documents, most of the ambiguities are reduced. Different ambiguities are detected and they are analyzed. This analysis is very helpful in reducing ambiguity. Thus, reduction of ambiguities in the early stage of the software development helps to develop high quality software. This leads to the satisfaction of the stakeholder. The identification and analyzes of the ambiguities

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

play a major role in the minimization of ambiguities in the SRS documents.

7. References:

[1] D. M. Berry, E. Kamsties, M. M. Krieger, From contract drafting to software specification: Linguistic sources of ambiguity, Technical Report, Technical Report, School of Computer Science, University of Waterloo 2003.

[2] L. Mich, M. Franch, P. L. Novi Inverardi, Market research for requirements analysis using linguistic tools, Requirements Engineering 9 (2004) 151-151.

[3] A. O. J. SABRIYE, W. M. N. W. ZAINON, An approach for detecting syntax and syntactic ambiguity in software requirement specification., journal of theoretical & applied information technology 96 (2018).

[4] M. Ceccato, N. Kiyavitskaya, N. Zeni, L. Mich, D. M. Berry, Ambiguity identification and measurement in natural language texts, Technical Report, University of Trento, 2004.

[5] B. Gleich, O. Creighton, L. Kof, Ambiguity detection: Towards a tool explaining ambiguity sources, in: International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, pp. 218-232.

[6] Prerana Chaithra, Surekha R., Shantharam Nayak “Influence of Mathematics on Software Engineering”, International Journal of Management Technology and Engineering, , ISSN: 2249-7455, Vol-9, Issue-5, May 2019, pp.3541-3547

[7] N. Kiyavitskaya, N. Zeni, L. Mich, D. M. Berry, Requirements for tools for ambiguity identification and measurement in natural language requirements specfications, Requirements engineering 13 (2008) 207-239.

[8] R. Garigliano, D. Nettleton, Neo-pragmatism, The LOLITA Project: the First Ten Years 3 (1997).

[9] R. G. Morgan, R. Garigliano, P. Callaghan, S. Poria, M. H. Smith, A. Urbanowicz, R. Collingham, M. Costantino, C. Cooper, L. Group, et al., University of durham: description of the lolita system as used in muc-6., in: Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995.

[10] R. Garigliano, A. Urbanowicz, D. J. Nettleton, Description of the Lolita system as used in muc-7, in: Proceedings of the 7th Message Understanding Conference (MUC{7), Citeseer.

[11] Prerana Chaithra, Shantharam Nayak, " Quality Assurance Techniques in SRS Documents”, Elsevier-Scopus indexed International Journal of Innovative Technology and Exploring Engineering (IJITEE), ISSN:2278-3075, Vol-9, Issue-2S, December 2019, pp. 14-18

[121] L. Mich, R. Garigliano, Nl-oops: A requirements analysis tool based on natural language processing, WIT Transactions on Information and Communication Technologies 28 (2002).

[13] W. M. Wilson, L. H. Rosenberg, L. E. Hyatt, Automated analysis of requirement specifications, in: Proceedings of the 19th international conference on Software engineering, pp. 161-171.

[14] L. Mich, R. Garigliano, et al., Ambiguity measures in requirement engineering, in: International Conference on Software Theory and Practice. ICS.

[15] Prerana Chaithra, Dr. Shantharam Nayak, "Machine Learning Technique for Identifying Ambiguities of in Software Requirements" Turkish Journal of Computer and Mathematics Education(TURCOMAT), Vol.12, No. 11 2021, pp 6852-6857

[16] F. Fabbrini, M. Fusani, S. Gnesi, G. Lami, The linguistic approach to the natural language requirements quality: benefit of the use of an automatic tool, in: Proceedings 26th Annual NASA Goddard Software Engineering Workshop, IEEE, pp. 97{105.

[17] M. Luisa, On the use of ambiguity measures in requirements analysis", in: Applications of Natural Language to Information Systems, 6th international workshop NLDB'01, Gesellschaft für Informatik e. V.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731