The Applications of Natural Language Processing (NLP) for Software Requirement Engineering - A Systematic Literature Review

Farhana Nazir[1], Wasi Haider Butt[2], Muhammad Waseem Anwar[3] and Muazzam A. Khan Khattak[4]

[1,2,3,4] College of Electrical and Mechanical Engineering (CEME), National University of Sciences and Technology (NUST), H-12 Islamabad, Pakistan.

farhana.nazir14@ce.ceme.edu.pk, wasi@ceme.nust.edu.pk,waseemanwar@ceme.nust.edu.pk, muazzamak@ceme.nust.edu.pk

Abstract. Natural Language Processing (NLP) is a well-known technique of artificial intelligence to extract the elements of concerns from raw plain text information. It can be utilized to process the early software requirements in order to achieve the goals like requirement prioritization and classification (functional and non-functional). To the best of our knowledge, no research work is available yet to examine and summarize the utilization of NLP in the domain of Software Requirement Engineering (SRE). Therefore, in this paper, we investigate the applications of NLP in the context of SRE. A Systematic Literature Review (SLR) is carried out to select 27 studies published during 2002-2016. Consequently, 6 NLP techniques and 14 existing tools are identified. Furthermore, 9 tools and 2 algorithms, proposed by the researchers, are presented. It has been concluded that the NLP techniques and tools are highly supportive to accelerate the SRE process. However, some manual operations are still required on initial plain text software requirements before applying the desired NLP techniques.

Keywords: NLP, SRE, NLP tools, Software requirements

1. Introduction

Software requirements are the foremost attributes of the system under development. These are usually classified into four major groups i.e. Business requirements, Functional Requirements (FR), Non-Functional Requirements (NFR) and Domain requirements. Initially, the software requirements are gathered and expressed in human readable natural language as a plain text. However, such textual requirements are of least use for technical stake holders. Therefore, it is essential to refine the early requirements for appropriate further utilization. However, manual enhancement of initial software requirement is laborious and time-consuming activity. On the other hand, Natural Language Processing (NLP) is a knowledge discovery approach to automatically extract the elements of concerns from raw plain text documents. Consequently, it is utilized to polish and extract desired software requirements from initial natural language artifacts.

As NLP provides sophisticated text mining features, it is commonly used in various software engineering areas [1-5]. For example, NLP is utilized to transform the functional software requirements into design artifacts [6-7]. It is also used to refine the ambiguities from initial textual requirements [8-10]. Although NLP techniques have been practiced in several software engineering areas [11-18], there is no study available yet to the best of our knowledge that investigate and summarize the applications of NLP in software requirement engineering domain. Therefore, in this article, we investigate the application of NLP techniques in software requirement engineering to get the answers of the following research questions:

RQ1: What are the leading software requirements areas where NLP techniques are frequently practiced?

RQ2: What are the primary NLP activities in the context of software requirement engineering?

RQ3: What are the leading tools, proposed / utilized by the researchers, for software requirement engineering?

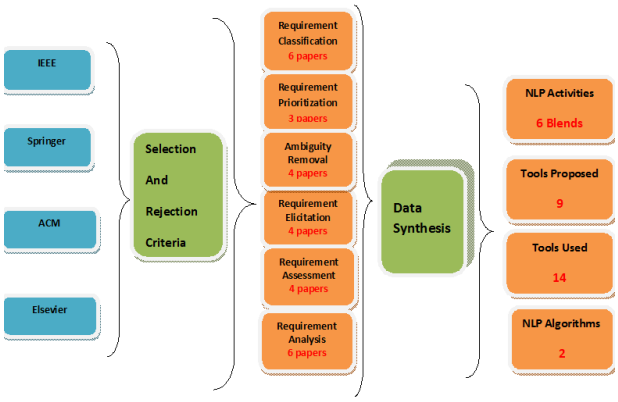

Fig. 1. Overview of Research

We develop a review protocol (Section **2.2**) that contains selection and rejection criteria. We define six categories (Section **2.1**) for the classification of selected 27 studies as shown in **Fig. 1**. We investigate the selected studies to identify 6 NLP techniques (Section **3.1**) that are frequently practiced independently as well as jointly in the area of software requirement engineering. Furthermore, we also identified 9 proposed tools, 14 utilized tools and 2 algorithms (Section **3.3**).

2. Research Methodology

Systematic Literature Review (SLR) [20] is used to perform this study. The research methodology consists of five stages: i) Category Definition ii) Selection Rejection Criteria iii) Search process iv) Quality Assessment v) Data Extraction

2.1 Category Definition

We define six significant categories for the selection of studies as follows:

Classification: The initial requirements are further classified on the basis of functionality like functional requirements, Non Functional Requirements etc. All research work dealing with such classification of requirements are included in this category.
Prioritization: In software system, priority is given to requirements according to the importance of their impact on the system. All the researches that deals with requirement prioritization are included in this category.
Ambiguity Removal: All research works that deal with the removal of ambiguous requirements from initial text are included in this category.
Requirement Elicitation: All research works that deal with the requirement elicitation from initial text by utilizing NLP techniques are placed in this category.
Requirement Assessment: The research works that deal with the evaluation of the impact of the requirements from initial plain text by employing NLP techniques are placed in this category.
Requirement Analysis: The studies that perform analysis on the initial textual requirements to get the desired features are placed in this category.
General: It is possible that some studies belong to more than one above-mentioned categories. All such studies are placed in this category.

2.2 Review Protocol

Review protocol has been set by maintaining the rules and regulations of SLR [20]. The RQ's and background are already covered in Section **1**. The details of further review protocol stages have been described in subsequent sections.

Selection and Rejection Criteria. The selection and rejection of research papers is based on following parameters:

- Select only those researches which are relevant to our research questions i.e. the study must utilize NLP approach for software requirement engineering.
- We only choose studies that must be published in one of these databases i.e. IEEE [21], ACM [22], Springer [23] and Elsevier [24] during 2002-2016.
- The papers with same research contents should be selected once.

Search Process. The given selection and rejection criteria shows that we just used four scientific databases (IEEE, ACM, Springer, and Elsevier) for our research process. We use specific terms related to our topic for the search and the results of these search are shown in **Table** 1. We apply various filters (e.g. publication year etc.) to shorten the number of search results.

Quality Assessment. We assess the quality of selected studies with following parameters: 1) The data assessment from selected studies is based on the solid facts and theoretical perspective 2) The selected studies have been properly validated through case studies and experiments 3) The search we choose is most important factor therefore we use four most genuine and globally accepted scientific databases i.e. IEEE, SPRINGER, ELSEVIER, ACM.

Table 1. Search terms and results

Sr.#	Terms	Operator	No of Search Results			
			IEEE	**ACM**	**Springer**	**Elsevier**
1	NLP		1925	1750	222	3,465
2	Software Requirement Classification	AND	213	401	61	1810
		OR	717	7,732	23	5,623
3	Software Requirement Categorization	AND	18	177	26	90
		OR	67	2,300	16	4,433
4	Software Requirement Classification +NLP	AND	11762	3	0	10
		OR	3,321	5,231	3,345	4,321

Data Collection and Synthesis. The elements of data extraction / synthesis are shown in **Table 2**. We use this template for each selected study to get desired data.

Table 2. Details of data collection and synthesis

Sr.#	Description	Details
1	References Information	Title, Author, Publication year, Publisher detail
Extraction of Data		
2	Overview	The main proposal / objective of study
3	Results	Results acquire from the study
Synthesis of Data		
4	Classification	According to defined categories **Table 3**
5	Techniques	NLP Techniques used in the studies **Table 4**
6	Tools	Tools used and proposed in studies (**Table 5** and **Table 6**)

3. Results

We have identified overall 27 research papers i.e. 6 Journal and 21 Conference. The selected studies are classified into six categories (Section **2.1**) as shown in **Table 3**.

Table 3. Classification of Studies

Sr.#	Category	Total	Corresponding Studies
1	Classification	6	[19][28][29][30][31]32]
2	Requirement prioritization	3	[33][34][35]
3	Ambiguity removal	4	[36][37][38][39]
4	Requirement elicitation	4	[40][41][42][43]
5	Quality Assessment	4	[44][45][46][47]
6	Requirement analysis	6	[48][49][50][51][52] [53]

3.1 NLP Techniques

We identify 6 main NLP techniques that have been utilized independently as well as jointly in the domain of software requirement engineering as shown in **Table 4**.

Table 4. Identification of NLP Techniques

Sr.#	NLP Techniques	Studies
1	Tokenization	[19][29][32][33][38][39][45][46]
2	POS Tagger	[19][29][30][31][32][33][34][36][37][38][44][46][52]
3	Text Chunking	[19][29][40][41][42][53]
4	Parsing	[19][32][44][45][48][50]
5	VSM	[19]
6	TF-IDF	[19][51][53]

There are studies that utilized more than one NLP activities e.g. [19] utilized all six NLP activities. Consequently, we place it against each technique as shown in **Table 4**. Similar is the case with other studies e.g. [30] etc.

3.3 Tools

We identified 14 existing NLP related tools that have been used by the researchers as shown in **Table 5.**

Table 5. Tools utilized in the given research context

Sr. No	Tool	Studies
1	SharpNLP [25], SMT Solver	[33]
2	C4.5 Decision tree algorithm ReqSAC, Rational XDE	[44]
3	NLP Engine	[19]
4	NARCIA (Natural Language Requirements Change Impact Analyzer)	[40]
5	Stanford tagger, NER.	[31]
6	NLTK (Natural Language ToolKit), PyEnchant	[45]
7	Drools Expert	[28]
8	Antiword, Jauman	[36]
9	C4.5 Algorithm	[47]
10	Stanford Parser	[32]
11	QUARS, ARM, WSD, RESI, SREE and NAI	[38]
12	RegeX Parser	[45]
13	GATE tool	[39]
14	OpenNLP	[53]

We identify 9 tools and 2 algorithms as given in **Table 6**.

Table 6. Proposed tools and algorithms

Sr. No	Tool / Algorithms Proposed	Reference
1	SNIPR	[33]
2	Text Classifier, FEATURE XTRACTOR	[44]
3	NARCIA	[42]
4	NLARE (Natural Lang. Automat. Requ. Evaluator)	[45]
5	WordNET	[36]
6	Model for NLP	[47]
7	MUPRET	[32]
8	ReqAligner	[46]
9	RUBRIC	[41]
Algorithms		
10	Unnamed Algorithm	[28]
11	LSAN Bayes classifier	[43]

4 Discussion and Limitations

It has been analysed that NLP techniques show encouraging outcomes while extracting relevant elements from initial plain text software requirements. However, it is usually required to perform few manual steps at lower level NLP activities e.g. tokenization and POS tagging. Therefore, it cannot be said that NLP fully automate the process of requirement refinement from initial plain text. However, the proposal of latest tools in this regard is highly beneficial. For example, [42] proposed a tool NARCIA for analysing the impact of change in natural language requirements. Although we utilized renowned scientific repositories, there is a chance that we might miss few studies from other scientific resources e.g. Google scholar etc.

5 Conclusion and Future work

This article investigates the applications of Natural Language Processing (NLP) for Software Requirement Engineering (SRE). A Systematic Literature Review (SLR) has been carried out to select 27 studies published during 2002-2016. As a result, 6 NLP techniques are identified that can be applied alone as well as jointly. Moreover, 14 existing tools are presented. Furthermore, the 9 tools and 2 algorithms, proposed by the researchers, are also identified. It has been concluded that the NLP techniques and tools certainly accelerate the SRE process. However, few manual operations are usually required on the initial plain text requirements before applying desired NLP approach. The tools and techniques, presented in this SLR, provide the platform for the SRE researchers. For example, this research can be extended to examine the application of NLP for the whole Software Development Life Cycle (SDLC) e.g. design and testing phases etc.

References

[1] K. Rayan, "the Role of Natural language in requirment Engineering," in *IEEE internation system requirement Engineering*, 1993.

[2] D. Binkley and D. Lawrie, "Information Retrieval Applications in Software Maintenance and Evolution," 2010.

[3] H. Funke, M., S. Sauer, and G. Engels B. Gu" ldali, "Semi- Automated Test Planning for e-ID Systems by Using Requirements Clustering," in *Automated Software Eng.*, 2009.

[4] W.B. Frakes and B.A. Nejmeh,.: SIGIR Forum, 1987, pp. vol. 21, pp. 30-36.

[5] D.M. Berry, and G.E. Kaiser Y.S. Maarek, "An Information Retrieval Approach for Automatically Constructing Software Libraries," in *IEEE Trans. Software Eng.*, 1991.

[6] L. Briand, and Y. Labiche T. Yue, "A Systematic Review of Transformation Approaches between User Requirements and Analysis Models," in *Requirements Eng.*, 2010, pp. 1-25.

[7] B. Paech and C.H. Martell, "Innovations for Requirement Analysis From Stakeholders' Needs to Formal Designs.," , 2008.

[8] N. Zeni, L. Mich D.M. Berry, "Requirements for Tools for Ambiguity Identification and Measurement in Natural Language Requirements Specifications," in *Req. Eng.*, 2008.

[9] B. Nuseibeh, A. de Roeck, and A. Willis C. Francis, "Identifying Nocuous Ambiguities in Natural Language Requirements," *14th IEEE Requirements Eng. Conf.*, 2006, pp. 59-68.

[10] A. Willis, Roeck, and B. H. Yang, "Automatic Detection of Nocuous Coordination Ambiguities in Natural Language Requirements," in *Proc.Automated Software Eng.*, 2010.

[11] J.H. Weber-Jahnke and A. Onabajo, "Finding Defects in Natural Language Confidentiality Requirements," in *Proc. 17th IEEE Int'l Requirements Eng. Conf.*, 2009.

[12] R. Oliveto, and P. Sgueglia De Lucia, "Incremental Approach and User Feedbacks: A Silver Bullet for Traceability Recovery," in *Proc. IEEE Software Maintenance*, 2006.

[13] R. Oliveto, and G. Tortora De Lucia, "Assessing IR-Based Traceability Recovery Tools through Controlled Experiments," in *Empirical Software Eng.*, 2009, pp. vol. 14 pp. 57-92.

[14] H. Sultanov and J.H. Hayes, "Application of Swarm Techniques to Requirements Engineering: Requirements Tracing," in *Proc. IEEE Int'l Requirements Eng. Conf.*, 2010.

[15] J.H. Hayes, A. Dekhtyar, and E.A. Holbrook S.K. Sundaram, "Assessing Traceability of Software Engineering Artifacts," in *Requirements Eng. J.*, 2010, pp. vol. 15, pp. 313-335.

[16] Duan and J. Cleland-Huang, "Clustering Support for Automated Tracing," in *Proc. 22nd IEEE/ACM Int'l Conf. Automated Software Eng.*, 2007.

[17] R. Settimi, and J. Cleland-Huang X. Zou, "Term-Based Enhancement Factors for Improving Automated Requirement Trace Retrieval," in *Proc. ACM Int'l Symp.* 2007.

[18] M. Lormans and A. van Deursen, "Can LSI Help Reconstructing Requirements Traceability in Design and Test?," in *Proc. Conf. Software Maintenance and Reeng.*, 2006.

[19] Giovanni Cantone and Gerardo Canfora Davide Falessi, "Empirical Principles and an Industrial Case Study in Retrieving Equivalent Requirementsvia Natural Language Processing Techniques," *IEEE TRANS. ON SOFTWARE ENGINEERING*, vol. 39, 2013.

[20] B., Kitechenhem, "Procedures for Performing Sysytematic Reviews," , Keele Univ. 2004.

[21] (2016) IEEE. [Online]. http://ieeexplore.ieee.org/

[22] ACM. [Online]. http://dl.acm.org/

[23] (2016) Springer Scientific Database. [Online]. http://www.springer.com/

[24] (2016) Elsevier. [Online]. https://www.elsevier.com

[25] CodePlex. [Online]. https://sharpnlp.codeplex.com/

[26] Torgier Dingsoyr Tore Dyba, "Empirical studies of agile software development: A systematic review," in *Elsevier Information and Software Technology*, 2008.

[27] M, Roberts, H, petticrew, "Sysytematic review in social sciences," , Maryland USA.

[28] Roshni R. , Shubhashis Vibhu Sharma, "A Framework for Identifying and Analyzing Non-functiona Requirements from Text," in *TwinPeaks Proc. of Req.*, 2014, pp. 1-8.

[29] Amjad Hudaib, Abushariah, Alqudah Fawaz Al-Zahgoul, "A suggested framework for Software Requirement Classification," in *IEEE 17th UKSIM-AMSS*, 2015.

[30] M.G. Ilieva and Olga Ormandjieva, "Automatic Transition of Natural Language Software Requirements Specification into Formal Presentation," , 2005.

[31] Ashish Sharma and Dharmender Singh, "Natural Language based Component Extraction from Requirement Engineering Document and its Complexity Analysis," , 2011.

[32] Thanwadee, Charnyote Assawamekin, "Ontology-based multiperspective requirements traceability framework," *Knowl. and Info.Sys.*, vol. Volume 25, no. 3, pp. 493–522, 2009.

[33] Shahryar, Thomas & McZara, "Software requirements prioritization and selection using linguistic tools and constraint solvers a controlled experiment," *Spr. Emp. Software Engineering*, vol. 20, no. 2015, pp. 1721-1766, 2014.

[34] Azlin Nordin and Keng-Yap Ng Kung-Kiu Lau, "Extracting Elements of Component-based Systems from Natural Language Requirements," , 2011.

[35] Muhammad Ramzan Shahbaz A. K. Ghayyur Muhammad Imran Babar, "Challenges and Future Trends in Software Requirements Prioritization," , 2011.

[36] Jin & LEPAGE, "Ambiguity Spotting using WordNet Semantic Similarity in Support to Recommended Practice for Software Requirements Specifications," in *(NLP-KE), 2011*

[37] Unnati S. Shah & Devesh C. Jinwala, "Resolving Ambiguities in Natural Language Software Requirements: A Comprehensive Survey," in *ACM SIGSOFT*, 2015.

[38] & Daniel M. Berry, & Erik Kamsties Christian Denger, "Higher Quality Requirements Specifications through Natural Language Patterns,". *SwSTE '03. Proceedings.*.

[39] Anuradha Chug, Allenoush Hayrapetian, Rajeev Raje Ruchika Malhotra, "Analyzing and Evaluating Security Features in Software Requirements," in *(ICICCS 2016)* , 2016.

[40] Mehrdad Sabetzadeh, Arda, Lionel C. Briand, Zimmer Arora, "Change Impact Analysis for Natural Language Requirements: An NLP Approach," *Req. Eng. Conf, 2015*.

[41] Mehrdad, Frank, Raul Gnaga, Lionel Briand Arora, "RUBRIC: A Flexible Tool for Automated Checking of Conformance to Requirement Boilerplates,"*Pro.Soft. Eng.*, 2013.

[42] Mehrdad Sabetzadeh,Arda, Lionel C., Frank Chetan, "NARCIA: An Automated Tool for Change Impact Analysis in Natural Language Requirements," in *Proc of Soft. Eng.*, 2015.

[43] Jianhua Guoa, Bing-Yi Jingc, Tieli Sunb Guozhong Fenga, "Feature Subset Selection Using Naive Bayes for Text Classification, 2015.

[44] Olga Ormandjieva and Leila Kosseim Hussain, "Automatic Quality Assessment of SRS Text by Means of a Decision-Tree-based Text Classifier," in *Quality Software Cong.* 2007.

[45] Reyes Juárez-Ramírez Carlos Huertas, "NLARE, A Natural Language Processing Tool for Automatic Requirements Evaluation," in *CUBE*, 2012, pp. 371-378.

[46] Claudia Marcos, J. Andres Diaz-Pace Alejandro Rago, "Identifying duplicate functionality in textual use cases by aligning semantic actions," in *Springer Softw Syst Model*, 2014.

[47] Ishrar Hussain, Kosseim Ormandjieva, "Toward a Text Classification System for the Quality Assessment of Software Requirements Written in Natural Language,", 2007.

[48] Dr Kyongho Min, Dr A.M. S.G. MacDonell, "AUTONOMOUS REQUIREMENTS SPECIFICATION PROCESSING USING NATURAL LANGUAGE PROCESSING,".

[49] Bjorn, Pa, Michael and Joachim Johan Natt Daga, "A Feasibility Study of Automated Natural Language Requirements Analysis in Market-Driven Development," , 2002.

[50] Selamawit,Lee Woon,H. H. ,Davor Erik, "NLP-KAOS for Systems Goal Elicitation:Smart Metering System Case Study," *IEEE Tran. on Soft. Eng.* , vol. 40, no. 10, Oct 2014.

[51] Giorgio O. Spagnolo, Felice Dell'Orletta Alessio Ferrari, "Mining Commonalities and Variabilities from Natural Language Documents," , 2013.

[52] Agung Fatwanto, "Software Requirements Specification Analysis Using Natural language Processing," in *QiR (Quality in Research), IEEE 2013 International Conference*, 2013.

[53] Jane Huffman Hayes, Alex Dekhtyar E. Ashlee Holbrook, "A study of methods for textual satisfaction assessment," *Emper. Software Enginnering*, vol. 18, no. 1, pp. 139-176, 2012.