

Crowd-Generated Data Mining for Continuous Requirements Elicitation

Ayed Alwadain¹

Computer Science Department. King Saud University
Riyadh, Saudi Arabia

Mishari Alshargi²

Information Systems Department. King Saud University
Master Degree Student

Abstract—In software development projects, the process of requirements engineering (RE) is one in which requirements are elicited, analyzed, documented, and managed. Requirements are traditionally collected using manual approaches, including interviews, surveys, and workshops. Employing traditional RE methods to engage a large base of users has always been a challenge, especially when the process involves users beyond the organization's reach. Furthermore, emerging software paradigms, such as mobile computing, social networks, and cloud computing, require better automated or semi-automated approaches for requirements elicitation because of the growth in systems users, the accessibility to crowd-generated data, and the rapid change of users' requirements. This research proposes a methodology to capture and analyze crowd-generated data (e.g., user feedback and comments) to find potential requirements for a software system in use. It semi-automates some requirements-elicitation tasks using data retrieval and natural language processing (NLP) techniques to extract potential requirements. It supports requirements engineers' efforts to gather potential requirements from crowd-generated data on social networks (e.g., Twitter). It is an assistive approach that taps into unused knowledge and experiences emphasizing continuous requirements elicitation during systems use.

Keywords—Requirements engineering; RE; crowd data mining; NLP; Twitter; continuous requirements elicitation

I. INTRODUCTION

Requirements engineering (RE) is the process of collecting, defining, documenting, and maintaining the requirements of a software system [1]. It is fundamental during the software development cycle to obtain users' needs by utilizing effective means of requirements elicitation, analysis, and management [2]. Getting the requirements right is important because mistakes cascade to subsequent development stages. Owing to poor RE practices, deficiencies at this phase cost more later and often result in systems failure [2-4].

Traditionally, elicitation is done at the beginning of software development. Recent approaches have advocated continuous requirement elicitation to capture user feedback and experiences during system's use [5]. Elicitation is needed during system's use to understand new feature requests, issues, and emerging requirements [6]. Requirements elicitation for traditional software systems has been well-studied, but new computing paradigms (e.g., social media, mobile apps, and cloud computing) require different assumptions and approaches [5]. These new computing paradigms enable users

to express their feedback and experiences online via social-network sites, forums, and blogs.

Because of changing contexts and user needs, continuous requirements elicitation should be adopted to ensure that requirements stay refreshed and that needs are addressed [7]. Stakeholder needs and technologies change over time, exacerbated by the rise of crowd-generated data. Automated requirements elicitation methods and analysis should be incorporated to enable requirements engineers to acquire and analyze online data efficiently. Automation facilitates access to online crowd-generated data and the use of these data for systems' improvements [7]. Automated or semi-automated requirements elicitation approaches should be able to overcome issues facing existing traditional approaches [8]. This research proposes a methodology that collects crowd-generated data from social networks (e.g., Twitter) and processes the data using natural language processing (NLP) techniques to extract potential emerging requirements for a certain software product.

The rest of the paper is organized as follows. Section 2 presents the literature review while Section 3 details the proposed methodology and its supporting tool. Sections 4 and 5 respectively present the discussion and the conclusion of this study.

II. LITERATURE REVIEW

The success of a system development or an upgrade depends on a well-developed RE process that successfully elicits and manages stakeholder requirements, resulting in a higher level of satisfaction [4, 9]. Requirements elicitation is traditionally the first phase of obtaining requirements. Elicitation is the most important phase because the collection of poor requirements can lead to project failure [10-12]. The involvement of users and customers in the RE process leads to many benefits, such as improved system acceptance, more accurate and complete requirements, and improved project success rates [13]. Many issues lead to poorly collected requirements, such as ambiguous project scopes, poor system understanding, and volatility where the evolved users' needs do not meet the original requirements [14].

Various requirements elicitation approaches have been suggested [15]. Most existing techniques are manual and assume the presence of the stakeholders involved. Employing such techniques can rapidly become expensive and resource-intensive, particularly when dealing with larger stakeholder populations [16-18]. Employing such techniques to engage a large user base has always been a challenge, especially when

there are large numbers of software users beyond the organization's reach [7]. Traditional RE approaches ignore opportunities to continuously engage large and heterogeneous groups of users who express their feedback on social networks and other websites. Better approaches are needed to tap crowd-generated data (e.g., feedback and opinions) to enable developers to consider them when developing their product's next version [7].

Stakeholder goals, environments, technology evolution, and the emergence of new computing paradigms require continuous requirements elicitation. For example, social-network sites and mobile applications generate data that can be collected and analyzed for potential requirements [7]. The rise of social networks and mobile applications has enabled the collection of massively generated crowd data. Social-network users can contribute their feedback directly or indirectly regarding system improvements [19, 20]. Whereas social networks were not designed for the purpose of requirements engineering, many companies include social networks in their software development process for this purpose [21].

Understanding public opinion and demands is a time-consuming process because of the high volume of crowd-generated data that must be reviewed [22]. Thus, automatic approaches to elicit and analyze such data are needed to achieve faster response times [7]. Automation facilitates the identification and analysis of potential requirements that are otherwise challenging and unreachable using traditional RE [8].

An emerging theme within RE research is Crowd-based requirements engineering (CrowdRE). It is an overarching term for the employment of automated or semi-automated methods to elicit and explore data from a crowd to derive potential requirements [7]. Crowdsourcing in requirements elicitation would enable the continuous requirements elicitation process during the life cycle of the software product. Such a practice would facilitate a deeper, wider, and more up-to-date perspective of how users perceive systems and to understand how requirements evolve [5]. Typically, a crowd is a large and heterogeneous group of existing or prospective users [7]. CrowdRE captures and analyzes user needs regarding the evolution of existing software systems, and it monitors software system usage and experiences. Crowd users report on a variety of aspects, such as problems, improvements, or extension ideas, which are useful for software development teams [7].

Crowd-generated textual data should be retrieved and processed with NLP techniques. NLP concerns the application of computational techniques for automatic parsing, analysis, and representation of human language. Many techniques have been suggested to process raw text in natural languages. For example, tokenization is a technique used for splitting a stream of text into its basic elements (i.e., tokens) such as words and phrases and other symbols [23]. Part-of-speech (POS) tagging is used to assign labels (e.g., noun, verb) to each identified token in a given text [24].

Several studies have attempted to automate the requirements elicitation process using NLP techniques. For example, NLP was used to extract early requirements matching

predefined patterns from user manuals and project reports. The text in these documents was tokenized and POS tagging was used to annotate the text. Then, topic modeling was applied to group-requirement items of similar content to avoid information overload. Whereas it is considered appropriate to reduce the burden of gathering requirements from scratch, some limitations have been reported, such as unclear extracted requirements and lack of comprehensive patterns [25].

Furthermore, an approach was developed to automate some requirements elicitation tasks using a tool that gathered stakeholder input in a centralized repository. Then, it used extended markup language and extensible stylesheet language transformations to render specifications [26]. In [27], a method was suggested to extract requirements from textual data in documents. NLP techniques (e.g., tokenization, POS tagging, and clustering) were used. Another study examined similar project documentation to extract potential requirements using NLP techniques (e.g., POS tagging) [28]. Another approach was proposed that used online customer reviews to extract needs and preferences regarding a specific product [29].

III. PROPOSED CONTINUOUS REQUIREMENTS-ELICITATION METHODOLOGY

This section outlines the proposed methodology and its supporting tool. This research provides an approach to automatically collect crowd-generated data via Twitter and process it using NLP techniques to find requirements. The proposed methodology is shown in Fig. 1. It enables engineers to elicit data from Twitter and analyze it using NLP techniques to find potential requirements. Twitter was selected because it is a popular microblogging social-media network and a potential data source to extract requirements [30, 31].

The methodology has four main steps: tweets collection and filtering, applying POS tagging, requirements generation, and requirements clustering. The following subsections illustrate the proposed methodology steps and its instantiation using AutoReq.

A. Tweet Collection (Pattern Matching) and Filtering

AutoReq enables requirements engineers to input a search keyword (e.g., the name of an existing system) and search twitter feed. The tool uses the Twitter application program interface (API) to retrieve real-time tweets matching the search criteria (i.e., predefined pattern). For example, if we were interested in finding the feedback of an existing system, X, the patterns added to AutoReq would include "X should ...," "X could ...," and "X lacks ...".

In this study, the software system of interest is Snapchat. It is a global multimedia messaging application. It was selected because it is widely used and has very diverse user groups with constantly evolving requirements. Prior to this experiment, we noticed users tweeting potential requirements, additional features, complaints, and other issues about Snapchat. An AutoReq pattern search list was used. Then, tweets were filtered from unwanted noise (e.g., hashtags, user mentions, and universal resource locators). They were then saved to the AutoReq database. During the active stream retrieval of tweets, more than 350 tweets having the word "Snapchat" were retrieved, and only 47 matched the predefined pattern.

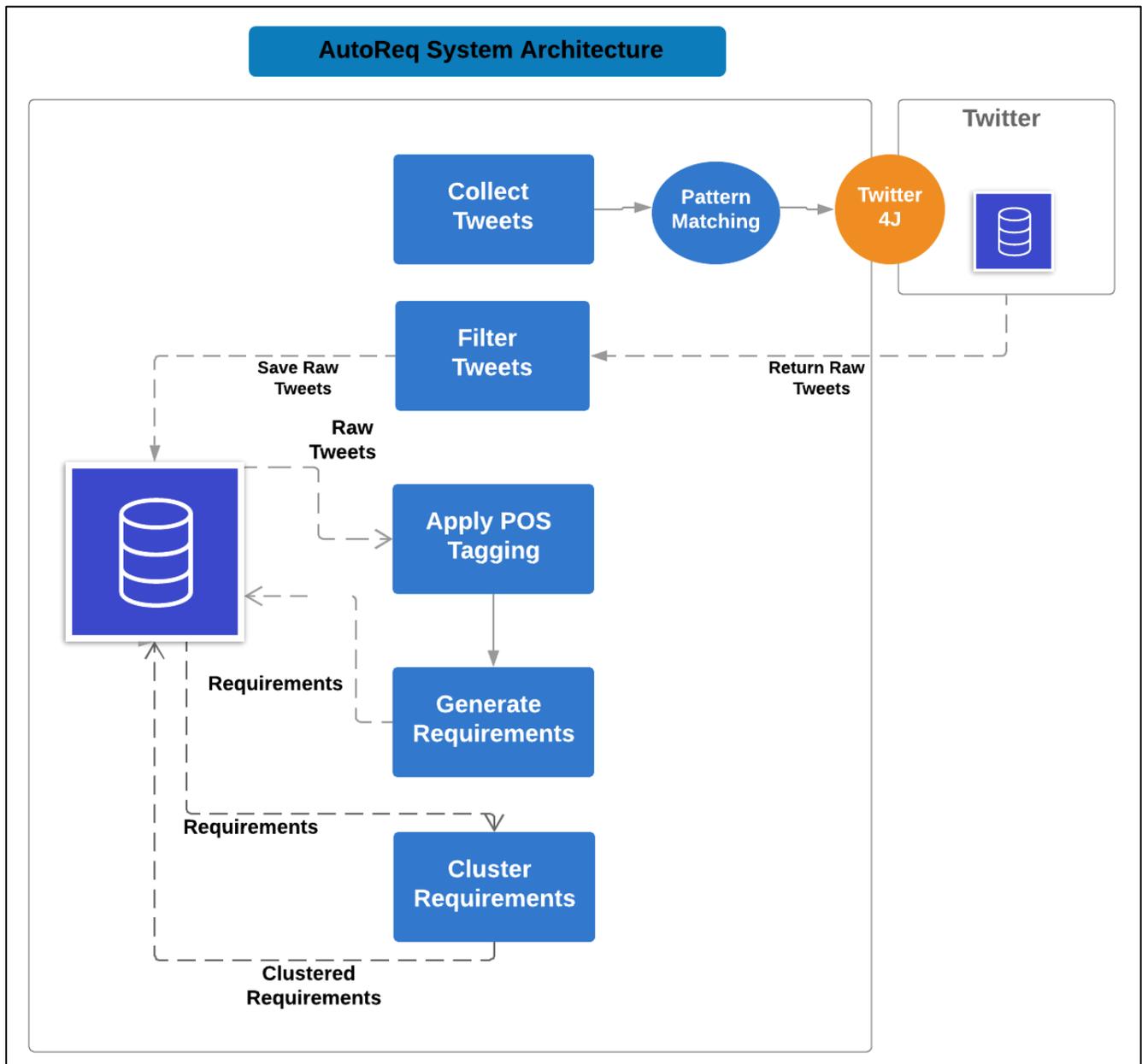


Fig. 1. Methodology and AutoReq System Architecture.

B. Part-of-Speech (POS) Tagging

Retrieved tweets were then tokenized by breaking them into tokens. Each tweet was then annotated using the Stanford POS tagger [24]. Tags were assigned to each word, depending on its role in the sentence. Still, there was some incorrect tagging. For example, the word “update” was incorrectly tagged in the tweet “snapchat should remove the last update.” It should have been tagged as a noun, but it was instead tagged as a verb. This phenomenon can lead to the generation of confusing requirements.

C. Requirements Generation

After tagging the words of each tweet, the first annotated verb and the closest three words were used to generate a requirement clause. Using a predefined requirement template within the tool, the requirement phrase was structured as “X shall + requirement clause.” In this experiment, generated requirements were structured as “Snapchat shall + requirement clause.” In some cases, a tweet contained more than one sentence. Thus, a recursion function of the tool was used to process the second part of the tweet. To find common conjunctions that potentially indicate the need for the use of the recursion capability, a qualitative analysis of the raw collected tweets was conducted. Then, the connection-words list was

developed based on the qualitative analysis and the use of existing conjunction words in English [32]. Using recursion, tweets were split into parts using a conjunction word. Each part of the tweet was processed alone, and then both parts were combined as one requirement using the format “Snapchat shall + combined tweet output.”

D. Requirements Clustering

After requirements generation, clustering can be useful, particularly in cases where the retrieved tweets are large. Generated requirements were clustered to provide an aggregated perspective of common themes from the generated requirements. Clustering was conducted using the RxNLP sentence-clustering API [33]. It groups text tokens on a sentence level. It can be applied to short texts, or, in this research, tweets, to build logical and meaningful clusters with suggested topics for each cluster.

Generated requirements were clustered based on the most frequent topic themes, making it easier to find requirements of interest. The results, as shown in Fig. 2, contain the cluster topic, cluster score, and cluster tweets. The cluster topic is a suggested name of the cluster contents, whereas the cluster score describes the topic meaningfulness and cluster size. It facilitates cluster ranking and unwanted cluster pruning.

```
Output x
Apache Tomcat or TomEE Log x Apache Tomcat or TomEE x MyNewProje
snapchat shall be full time jobs
=====
Topics are: [snap:3.69]
Snapchat shall make group snap to that whole groups
Snapchat shall show time person snap
Snapchat shall give notification when someone opens snap
snapchat shall do to where look previous snap
=====
Topics are: [group:3.32]
Snapchat shall make group snap to that whole groups
Snapchat shall have group
=====
Topics are: [everyon:3.32]
Snapchat shall bring able to see everyones best
Snapchat shall take bunny filter to everyone
snapchat shall be filter that turned everyone beauty queen
=====
Topics are: [stori:3.32, limit:2.52]
Snapchat shall have story limit
Snapchat shall have limit how many stories person
Snapchat shall tell how many times someone has story
=====
Topics are: [chang usernam:2.71, usernam:2.52]
Snapchat shall have where change username
snapchat shall allow to change username
=====
```

Fig. 2. Requirements Clustering.

IV. DISCUSSION

RE mostly uses traditional data sources (e.g., forms, reports, notes, workshops, and meetings) and manual approaches such as interviews for capturing stakeholder requirements. The wide use of social networks and mobile apps has contributed to a massive growth in online crowd-generated data. Crowd users report on a variety of issues based on software problems, desired improvements, and extension ideas, which are potentially useful for software development teams [7]. These data are often massive, unstructured, and manually inaccessible [22]. Hence, recent research has called for the development of automated approaches to capture and analyze these data to locate potential requirements. A rising opportunity for RE lies within the use of hidden and unused crowd-generated data [7].

This research endeavored to explore this research area and contributed as follows. First, this study is early research exploring the use of crowd-generated social-networks data to find new requirements for an existing software system. It proposed a methodology and a tool to capture and analyze crowd-generated data to identify potential requirements. Such an approach is needed to achieve fast responses to user needs and to explore the hidden, unused data generated by users on social networks [7]. The developed methodology and tool support requirements engineers in their tasks of monitoring and eliciting potential requirements from crowd-generated data using their reported feedback, comments, and experiences.

Second, this research used NLP techniques to automatically analyze the captured textual crowd-generated data (i.e., tweets). NLP techniques support requirements engineers by automating parsing, analysis, and representation of textual data. Manual inspection, filtering, and processing are time-consuming and resource-intensive. Thus, an automated approach of crowd-generated data retrieval and processing reduces time and resource utilization. Nonetheless, there were some issues with unclear generated requirements phrases from incorrect tagging when using a POS tagger. To overcome this, the developed tool was designed to show the original tweets and the generated associated requirement phrases to help requirements engineers trace and understand the generated requirements.

Third, this research emphasized the continuous requirements elicitation process over a software product life cycle using crowd-generated data [5]. Feedback and experiences of current or prospective users were continuously captured about new features, emerging needs, and other issues. This approach is not easily implementable with traditional data sources, such as manuals and reports.

Fourth, a sentence-clustering technique was used to cluster requirements based on their similarity [33]. In this research, every processed tweet was treated as a unique requirement and a genuine idea that may lead to redundant requirements. Thus, a sentence-clustering technique was used to enable requirements engineers to look at clusters when the generated requirements are large. This reduces requirements engineers' manual efforts. Previous research mostly used topic modeling

techniques to detect the most frequent words in their data source to build requirements [25]. Some studies used clustering to cluster the requirements based on predefined centroids, regardless of similarity [27].

V. CONCLUSION

Requirements elicitation is a crucial phase in the software development life cycle designed to fully understand users' needs. During the elicitation process, interviews, workshops, reports, and manuals are typically used to generate requirements. However, emerging computing paradigms and the massive growth of crowd-generated data require automated elicitation approaches. Crowds directly or indirectly express their feedback, comments, and opinions regarding an existing system on social networks and similar platforms. Gathering data using existing requirements elicitation techniques is an arduous process, particularly when dealing with large-scale systems.

This research proposed a methodology and proof-of-concept to automate the retrieval and analysis of crowd-generated data from Twitter using NLP techniques to find potential requirements of an existing software product. This is an early study investigating the use of crowd-generated data to find potential requirements. It employs NLP techniques to automatically analyze captured textual data, and it enables a continuous requirements elicitation process during the use of software products. It also uses a clustering-sentence technique to cluster requirements based on their similarity to automate grouping of similar tweets. This reduces manual RE efforts.

Because every research effort is limited, there are some limitations with this study. First, because we proposed a semi-automated tool, there needs to be an RE verification and evaluation of the generated requirements to assess their relevance and importance. Second, this study inherited some limitations of the applied NLP techniques, particularly POS tagging. In addition, automated text processing and analysis have their own limitations. For example, there were some generated requirements that were not meaningful because of either incorrect tagging or retrieval. Another limitation was inherited from the data source, owing restricted access to tweets using the Twitter API and the 140-character limitations at the time of the execution of the experiment.

In the future, extra efforts are needed to improve the suggested approach. For example, additional NLP techniques (e.g., collaborative filtering) should be included to extract relevant requirements. Furthermore, richer data sources are suggested, including Facebook and online app reviews, to collect richer requirements. In general, further research is needed to develop methods and tools that facilitate continuous requirements elicitation to retrieve and analyze online crowd-generated data during software systems use.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this research.

REFERENCES

- [1] S. Gupta and M. Wadhwa, "Requirement Engineering: An Overview," *International Journal of Research and Engineering*, vol. 1, pp. 155-160, 2013.
- [2] J. Vijayan and G. Raju, "A New Approach to Requirements Elicitation Using Paper Prototype," *International Journal of Advanced Science and Technology*, vol. 28, pp. 9-16, 2011.
- [3] P. Rajagopal, R. Lee, T. Ahlswede, C. Chia-Chu, and D. Karolak, "A new approach for software requirements elicitation," in *Proceedings of the 6th IEEE International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2005.
- [4] D. Pandey and V. Pandey, "Requirement Engineering: An Approach to Quality Software Development," *Journal of Global Research in Computer Science*, vol. 3, pp. 31-33, 2012.
- [5] M. Hosseini, K. Phalp, J. Taylor, and R. Ali, "Towards Crowdsourcing for Requirements Engineering," in *The 20th International Working Conference on Requirements Engineering: Foundation for Software Quality*, 2014.
- [6] J. A. Khan, L. Liu, L. Wen, and R. Ali, "Crowd Intelligence in Requirements Engineering: Current Status and Future Directions," in *Requirements Engineering: Foundation for Software Quality*, 2019, pp. 245-261.
- [7] E. C. Groen, N. Seyff, R. Ali, F. Dalpiaz, J. Doerr, E. Guzman, et al., "The Crowd in Requirements Engineering: The Landscape and Challenges," *IEEE Software*, vol. 34, pp. 44-52, 2017.
- [8] N. Mulla and S. Girase, "A New Approach to Requirement Elicitation Based on Stakeholder Recommendation and Collaborative Filtering," *International Journal of Software Engineering & Applications (IJSEA)*, vol. 3, pp. 51-60, 2012.
- [9] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," in *Proceedings of the Conference on the Future of Software Engineering*, 2000, pp. 35-46.
- [10] S. Khan, A. B. Dulloo, and M. Verma, "Systematic Review of Requirement Elicitation Techniques," *International Journal of Information and Computation Technology*, vol. 4, pp. 133-138, 2014.
- [11] O. I. A. Mrayat, N. M. Norwawi, and N. Basir, "Requirements Elicitation Techniques: Comparative Study," *International Journal of Recent Development in Engineering and Technology*, vol. 1, pp. 1-10, 2013.
- [12] S. Sharma and S. Pandey, "Revisiting Requirements Elicitation Techniques," *International Journal of Computer Applications*, vol. 75, pp. 35-39, 2013.
- [13] R. Snijders, Ö. Atilla, F. Dalpiaz, and S. Brinkkemper, "Crowd-centric requirements engineering: A method based on crowdsourcing and gamification," *Master's Thesis*, Utrecht University, 2015.
- [14] M. G. Christel and K. C. Kang, "Issues in requirements elicitation," *Technical Report CMU/SEI-92-TR-012*, Software Eng. Inst., Carnegie Mellon University, 1992.
- [15] M. S. Tabbassum Iqbal, "Requirement Elicitation Technique: - A Review Paper," *International Journal of Computer & Mathematical Sciences*, vol. 3, pp. 1-6, 2014.
- [16] M. Yousuf, M. Asger, and M. U. Bokhari, "A Systematic Approach for Requirements Elicitation Techniques Selection: A Review," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, pp. 1399-1403, 2015.
- [17] K. Wnuk, "Understanding and supporting large-scale requirements management," *Licentiate Thesis*, Department of Computer Science, Lund University, vol. 2010, 2010.
- [18] U. Sajjad and M. Q. Hanif, "Issues and challenges of requirement elicitation in large web projects," *School of Computing*, Blekinge Institute of Technology, Ronneby, Sweden, 2010.
- [19] D. T. Nguyen, N. P. Nguyen, and M. T. Thai, "Sources of misinformation in Online Social Networks: Who to suspect?," presented at the *Military Communications Conference*, 2012.
- [20] D. S. Kim and J. W. Kim, "Public Opinion Sensing and Trend Analysis on Social Media: A Study on Nuclear Power on Twitter," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 9, pp. 373-384, 2014.
- [21] N. Seyff, I. Todoran, K. Caluser, L. Singer, and M. Glinz, "Using Popular Social Network Sites to Support Requirements Elicitation, Prioritization and Negotiation," *Journal of Internet Services and Applications*, vol. 6, pp. 1-16, 2015.
- [22] M. Yousuf and M. Asger, "Comparison of Various Requirements Elicitation Techniques," *International Journal of Computer Applications*, vol. 116, pp. 8-15, 2015.
- [23] T. Verma and D. G. Renu, "Tokenization and Filtering Process in RapidMiner," *International Journal of Applied Information Systems (IJ AIS)–ISSN*, vol. 7, pp. 2249-0868, 2014.
- [24] K. Toutanova, D. Klein, C. Manning, and Y. Singer, "Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003.
- [25] Y. Li, E. Guzman, K. Tsiamoura, F. Schneider, and B. Bruegge, "Automated Requirements Extraction for Scientific Software," *Procedia Computer Science*, vol. 51, pp. 582-591, 2015.
- [26] N. W. Kassel and B. A. Malloy, "An approach to automate requirements elicitation and specification," in *International Conference Software Engineering and Applications*, 2003.
- [27] S. Murugesu and A. Jaya, "A Generic Framework for Requirements Elicitation from Informal Descriptions," *International Journal of Advanced Research in Computer Engineering & Technology*, vol. 3, pp. 2545-2549, 2014.
- [28] K. Li, R. Dewar, and R. Pooley, "Requirements capture in natural language problem statements," *Heriot-Watt Technical Report HW-MACS-TR-0023*, 2004.
- [29] R. Rai, "Identifying key product attributes and their importance levels from online customer reviews," in *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE2011)*, Paper No. DETC2012-70493, 2012.
- [30] S. Arapostathis and S. Kalogirou, "Twitter data as a volunteered geographic information source: review paper of recent research analysis methods and applications," in *Proceedings of the 1st Spatial Analysis Conference*, 2013.
- [31] Y.-k. Lee, N.-H. Kim, D. Kim, D.-h. Lee, and H. P. In, "Customer Requirements Elicitation Based on Social Network Service," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 5, pp. 1733-1750, 2011.
- [32] Smart Words. (20 Feb). Conjunctions. Available: <https://www.smart-words.org/linking-words/conjunctions.html>
- [33] RxNLP. Sentence Clustering API. Available: <http://www.rxnlp.com/api-reference/cluster-sentences-api-reference/>