Machine learning in requirements elicitation: a literature review

Cheligeer Cheligeer[1], Jingwei Huang[2], Guosong Wu[3], Nadia Bhuiyan[4], Yuan Xu[3] and Yong Zeng[1] (iD)

[1]Concordia Institute for Information Systems Engineering, Concordia University, Montreal, QC, Canada; [2]Engineering Management & Systems Engineering, Old Dominion University, Norfolk, VA, USA; [3]Center for Health Informatics, Community Health Sciences, Cumming School of Medicine, University of Calgary, Calgary, AB, Canada and [4]Department of Mechanical, Industrial and Aerospace Engineering, Concordia University, Montreal, QC, Canada

Abstract

A growing trend in requirements elicitation is the use of machine learning (ML) techniques to automate the cumbersome requirement handling process. This literature review summarizes and analyzes studies that incorporate ML and natural language processing (NLP) into demand elicitation. We answer the following research questions: (1) What requirement elicitation activities are supported by ML? (2) What data sources are used to build ML-based requirement solutions? (3) What technologies, algorithms, and tools are used to build ML-based requirement elicitation? (4) How to construct an ML-based requirements elicitation method? (5) What are the available tools to support ML-based requirements elicitation methodology? Keywords derived from these research questions led to 975 records initially retrieved from 7 scientific search engines. Finally, 86 articles were selected for inclusion in the review. As the primary research finding, we identified 15 ML-based requirement elicitation tasks and classified them into four categories. Twelve different data sources for building a data-driven model are identified and classified in this literature review. In addition, we categorized the techniques for constructing ML-based requirement elicitation methods into five parts, which are *Data Cleansing and Preprocessing, Textual Feature Extraction, Learning, Evaluation, and Tools.* More specifically, 3 categories of preprocessing methods, 3 different feature extraction strategies, 12 different families of learning methods, 2 different evaluation strategies, and various off-the-shelf publicly available tools were identified. Furthermore, we discussed the limitations of the current studies and proposed eight potential directions for future research.

Introduction

Requirement elicitation is one of the important processes of product development. Several conventional requirement elicitation techniques, such as interviews, meetings, and brainstorming, are used to collect precise and individualized requirements. However, due to the ever-growing demands of end-users and the rapid pace of product iterations, the use of only traditional methods to elicit requirements would be quite insufficient.

The fourth industrial revolution is triggering a pervasive digital transformation in many fields of human activities. Particularly, engineering is being transformed into "Digital Engineering" (Zimmerman, 2017; US DoD, 2018). In digital engineering, digital data and models will be shared in the engineering life cycle (US DoD, 2018); engineering artifacts and processes will be digitalized with standardized digital representation, unique identifier, and the augmented metadata about their attributes, including provenance, thus making those digital artifacts machine-processible, uniquely identifiable, traceable, and accountable (Huang *et al.*, 2020). The digital engineering transformation brings both opportunities and challenges for requirements elicitation.

The evolution of digital transformation has led to improved productivity, quality, and customer satisfaction through agile and robust big data collection, analysis, learning, and decision-making processes. Success stories, advancing technologies, and growing customer demands are why digitalization has become necessary for various fields. For example, in recent years, there has been a growing number of studies involving a digital transformation in requirement engineering, such as identifying requirements from documents (Wang *et al.*, 2019), automatically classifying the requirements (Casamayor *et al.*, 2012), and prioritizing the requirements (Maiti and Mitropoulos, 2017). By applying advanced technologies and shifting the existing process to a new digitized paradigm, it may be possible to solve the problem.

Traditionally, expert experience or intuition has been used to direct requirement elicitation activities. Each decision is based on a combination of implicit and explicit domain expertise (Maalej and Thurimella, 2013). Developing a computer model that mimics expert reasoning

CrossMark

with knowledge is expensive to construct and maintain. A data-driven strategy, unlike knowledge-based systems, does not require codifying the rules and knowledge for decision-making. The term *data-driven* refers to a decision-making strategy based on data analytics, interpretation, and prediction rather than pure intuition (Provost and Fawcett, 2013). Over the past 15 years, several studies have been published on the application of machine learning (ML) to requirements engineering, followed by reviews that summarize these studies (Meth *et al.*, 2013; Wong *et al.*, 2017; Lim *et al.*, 2021). Different from those existing studies, this literature review includes 86 studies from 2007 to the present, and categorizes the included works from 7 perspectives, including tasks, data collection, data cleansing and preprocessing, textual feature extraction, learning, evaluation, and the open-source tools.

The rest of this paper is structured as follows. In Section "Related works", literature reviews related to the proposed review are summarized; and Section "Review methodology", the scope and methodology of the literature review, as well as search strategies, criteria for inclusion and exclusion, and the data extraction template, are presented. Section "Results" shows the primary results of the literature review. Section "Findings" summarizes the major findings from the review by analyzing the included works and categorizing them into various categories from seven different research concerns. In Section "Open issues and future works", the current role of ML in requirement elicitation and its limitations are discussed. In addition, the open issues and potential future works in this field are discussed. In Section "Limitation of this review", we discuss the potential threat to validity of the review and the measures we took to address these limitations. Finally, Section "Conclusion" concludes the paper.

Related works

To our knowledge, eight existing review articles, as shown in Table 1, are relevant to our study. Meth *et al.* (2013) conducted a review mainly focused on the automated approach applied for requirements elicitation, mainly focusing on the degree of the automation of proposed approaches. Binkhonain and Zhao (2019) introduced ML algorithms in the requirements elicitation domain by dividing the 24 related articles into 3 sections: NLP techniques, ML algorithms, and evaluation. Perez-Verdejo *et al.* (2020) applied topic models and visualization techniques to analyze ML-based requirement classification articles. Wong *et al.* (2017) identified various software requirements elicitation methods, including manual, rule-based, and ML-based approaches. Shabestari *et al.* (2019) proposed a systematic literature review that covers early product development phases, including various activities such as requirements elicitation, requirement identification, and requirement categorization. Similarly, Sampada *et al.* (2020) focus on the early requirement phases but are more concerned with requirements elicitation and documentation. Ahmad *et al.* (2020) reviewed a collection of articles for identifying requirements for Q&A platforms.

Among the existing studies, one existing work proposed by Lim *et al.* (2021) is the closest to our research, which was conducted almost concurrently with ours. Both works aim to introduce the current state of the works in data-driven requirements elicitation; however, the focuses of the two works are different. Lim *et al.* (2021) focus more on data sources, data types, learning techniques, and degree of automation. In comparison, the present review focuses more on technical details, such as text features. Our

work aims to provide a comprehensive overview of current work and include a more detailed investigation into the types of requirement elicitation tasks, existing methods, algorithms, and tools. This review could provide a more practical guide to requirements elicitation researchers, and engineers to leverage the existing techniques in their projects.

Review methodology

Review scope

The review adheres to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) reporting structure, aiming to answer the five research questions below:

RQ1. What requirement elicitation activities are supported by ML?
RQ2. What data sources are used to build ML-based requirement solutions?
RQ3. What technologies and algorithms are used to build ML-based requirement elicitation?
RQ4. What are the available tools to support ML-based requirements elicitation methodology?
RQ5. How to construct an ML-based requirements elicitation method?

The review scope is defined in Table 2 according to Cooper's taxonomy for literature review, including focus, goal perspective, coverage, organization, and audience (Cooper, 1988; Cooper *et al.*, 2019). First, in this work, the emphasis is on practical solutions that can be applied; therefore, the theoretical works are not our *focus*. Second, this study aims to synthesize and integrate existing studies to identify the specific requirement elicitation tasks supported by ML; thus, criticism of the field or related works is not a *goal* of this article. Third, this paper does not take an espoused *perspective* to advocate for or against ML-based requirements elicitation. Instead, it demonstrates how the existing work would convert requirements elicitation challenges into ML problems. Fourth, the *coverage* of the literature is a non-exhaustive set of research articles that are retrieved by search queries and filtered by inclusion and exclusion criteria. Fifth, the work applies a methodological *organization* that group and organize similar methodologies or tools together, presenting a modular organization to the target audience. Finally, the targeted *audiences* are mainly requirements analysts, engineers, and scholars.

Databases and search strategies

Seven bibliographic databases, including Scopus, Web of Science, Google Scholar, IEEE Xplore, Springer Link, ACM digital library, and ASME digital library are adopted to guarantee the coverage of the review. Three search strategies are applied: (1) the query expanding strategy is used to add synonyms, inflectional, and derivational morphological forms to the original term; (2) a wildcard character is used to capture multiple forms of a keyword by replacing one or more characters with a star symbol (*) or question marks (?); and (3) a query scoping strategy is applied when the search term is too general to retrieve a related result, such as adding terms "system engineering" or "requirement engineering" in addition to the search string.

Table 1. Related works

Title	Reference	Included works	The latest work reviewed	Main focus
The state of the art in automated requirements elicitation	Meth *et al.* (2013)	36	2010	The automated approach in requirements elicitation; degree of automation; knowledge reuse; evaluation; and the relationship between the included works.
A systematic literature review about software requirements elicitation	Wong *et al.* (2017)	42	2014	Level of automation; knowledge reuse; the importance of human factors; collaborative approach; and types of projects.
A review of ML algorithms for identification and classification of non-functional requirements	Binkhonain and Zhao (2019)	24	2017	ML algorithms for non-function requirements identification/classification.
A survey on the applications of ML in the early phases of product development	Shabestari *et al.* (2019)	40	2018	ML in requirements, modeling, and concept design.
A systematic literature review on using ML algorithms for software requirements identification on stack overflow	Ahmad *et al.* (2020)	12	2018	The ML and NLP techniques were applied for the required identification task on stack overflow data.
A review on advanced techniques of requirements elicitation and specification in software development stages	Sampada *et al.* (2020)	13	2019	Introduced the application of NLP techniques and ML algorithms in requirements elicitation and specification activities.
A systematic literature review on ML for automated requirements classification	Perez-Verdejo *et al.* (2020)	13	2019	The ML algorithms are used in requirement classification tasks.
Data-driven requirements elicitation: a systematic literature review	Lim *et al.* (2021)	68	2020	Data; ML techniques; evaluation methods; degree of automation in requirements elicitation.

Based on the above consideration, the search strings are defined as follows:

("Requirement" OR "Demand" OR "Need") AND ("Elicit" OR "Collect" OR "Gather" OR "Detect" OR "Identify" OR "Classify") AND ("ML" OR "Machine Learning" OR "Deep Learning" OR "Text Mining" OR "Data Mining" OR "NLP" OR "Natural Language Processing" OR "Neural Network" OR "Automated" OR "Data-driven" OR "Decision Support").

Due to search functions being different across the seven academic search engines, in the actual search pattern certain differences may exist. For example, the Web of Science engine supports additional Boolean operators, such as the "*NEAR*" operator that provides additional restrictions on the "AND" operator by considering a fixed-size context window.

Inclusion exclusion criteria and paper screening

The next step of the literature review is selecting studies by screening the title, abstract, and full text of the works found in the previous steps. We applied the inclusion/exclusion criteria in Table 3.

Table 2. The research scope under Cooper's literature review taxonomy

Characteristic	Categories	The focus of this work
Focus	Research findings	√
	Research methods	√
	Practices of applications	√
	Theories	×
Goal	Integration	√
	Identification of the central issue	√
	Criticism	×
Perspective	Neutral representation	√
	Espousal of position	×
Coverage	Exhaustive with selective citation	√
	Exhaustive	×
	Representative	×
	Central or pivotal	×
Organization	Methodological	√
	Conceptual	×
	Historical	×
Audience	Specialized scholars	√
	General scholars	√
	Practitioners or policymakers	×
	General public	×

Table 3. Inclusion/exclusion criteria

Exclusion	Inclusion
The work is a survey paper or literature review.	The work should either capture requirements in an automated way from text data or support the requirement elicitation process with an automated approach trained from big data.
The work is an editorial, conference abstract, or an introductory popular science article.	The article presents the details of the datasets and data processing.
The work only generally describes the problem and the solution without providing an experiment result or convincing evidence.	The article explains in detail the ML algorithm that they employed, such as the learning algorithms they used and the validation strategy they employed.
If there are multiple similar works from the same authors, only one of the earliest works would be retained.	The full text of the article should be accessible.
The work is written in a non-English language.	The article is written in English.

Data extraction table

Research information was collected from each included article with a data extraction form. Basic information about the study (author, title, year of publication, etc.) and content related to research interests (data source, preprocess, feature extraction, etc.) was collected. This includes 14 data elements described in Table 4. The required data fields are designed as open questions, which require reviewers to collect, summarize, and categorize data from the collect works.

Based on the search strategies applied on the 7 included scientific search engines, 975 papers were retrieved. Upon initial screening and the title screening, 915 works were forwarded to the title-abstract screening. A subset of 774 was irrelevant and thus discarded. As a result, 129 papers are retained for the full-text screening. In accordance with the inclusion–exclusion criteria, 43 articles were excluded, and finally, 86 articles were selected. The complete process of study selection is shown in Figure 1.

Table 4. Elements of data extraction table

| #	Data	Description
1	*Author(s)*	Author(s) of the included work in the literature review.
2	*Title*	Title of the included work in the literature review.
3	*Country*	The country of the corresponding author is the primary country, and the rest of the countries follow the order of the author list.
4	*Year*	The published year of the work.
5	*Citation count*	The number of citations of included work. (From Google Scholar, until 2021-10-20)
6	*Venue*	The publication type of the work includes conference, journal, book chapter, and workshop.
7	*Venue title*	The name of the journal or conference contains the article.
8	*Requirements elicitation task(s)*	Identify which requirements elicitation subtask is supported by the paper.
9	*Data source*	The data source is used for training or validation purposes.
10	*Preprocess*	The specific data preprocessing techniques from the selected paper.
11	*Feature extraction*	The specific features applied by the selected work in the literature review.
12	*Learning algorithm*	The learning algorithm was adopted by the included study in the literature review.
13	*Evaluation method*	The evolution method was introduced by the selected papers in the literature review.
14	*Tools*	List the tools mentioned by the authors in the paper as aids to their work.

Results

Tendencies of the publications

The overall trend for the 86 articles is shown in Figure 2, and an increasing trend can be observed. The reviewed studies came from 30 different countries, and 17 of them were conducted in more than one country. The average number of publications in each country is 3.8, with seven countries having more publications than the average, which are the United States ($n = 28$, 25.0%), China ($n = 14$, 12.5%), Germany ($n = 13$, 11.6%), Canada ($n = 8$, 7.1%), Singapore ($n = 5$, 4.5%), South Korea ($n = 5$, 4.5%), and the United Kingdom ($n = 5$, 4.5%).

Thirty-nine of the studies are conference papers ($n = 39$, 45.3%) and 31 are journal papers ($n = 31$, 36.0%). In addition, eight workshop papers ($n = 8$, 9.3%) and eight book sections ($n = 8$, 9.3%) are included, respectively. The included conference papers are collected in 23 unique conference proceedings, with 16 works appearing in the Proceedings of the *IEEE*
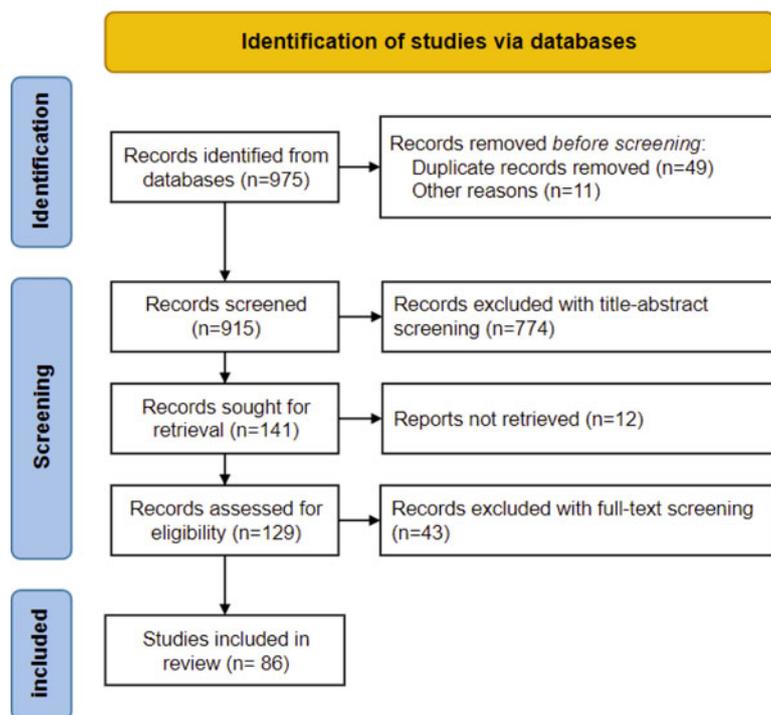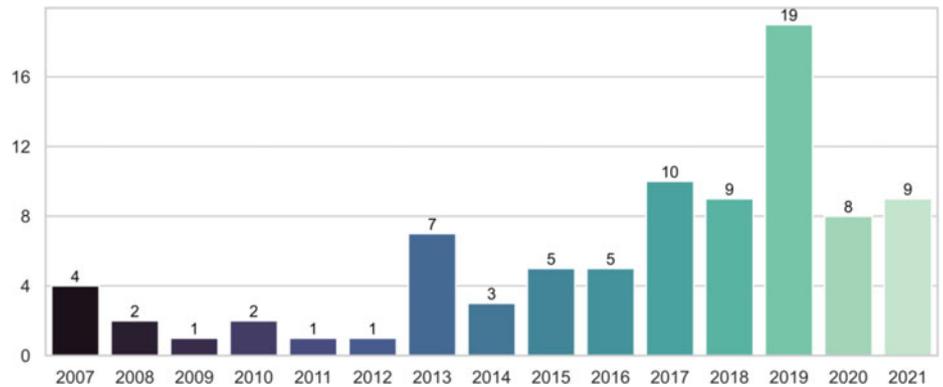

Fig. 1. PRISMA flowchart.

Fig. 2. The number of included papers by year.

International Conference on Requirements Engineering. The majority of the journal publications in this collection are from the *Journal of Mechanical Design, Information and Software Technology*, and *Requirement Engineering*.

The data source for ML-based requirement elicitation

The requirement specification (RS) is a textual document that systematically stores system definition and system requirements (Bourque and Fairley, 2014). In the reviewed articles, 17 works applied available requirement specifications to build ML-based solutions to support requirement elicitation. The majority (*n* = 12) of the requirements specifications in the selected studies are written in English, and two of the requirement specification are bilingual (Ko *et al.*, 2007; Lyutov *et al.*, 2019), three are written in the non-English language (Falessi *et al.*, 2010; Ott, 2013; Gulle *et al.*, 2020).

A total of 27 works are based on already existing corpora, mainly DePaul's NFR corpus (Cleland-Huang *et al.*, 2007), SecReq dataset (Knauss *et al.*, 2011), and PURE dataset (Ferrari *et al.*, 2017). The DePaul's NFR corpus and SecReq dataset are labeled datasets for specified tasks and the PURE corpus is unlabeled that contains multiple raw requirement documents.

About half of the included works applied available data from user-generated content (UGC) to aid requirement elicitation. The UGC includes textual data from e-commerce websites (*n* = 16), mobile app marketplace (*n* = 13), microblogs and social media (*n* = 5), organizational internal forum (Lange, 2008), GitHub repository (Nyamawe *et al.*, 2019), and crowdsourcing platforms (Jones and Kim, 2015).

Requirement data preprocessing

On average, most papers described at least one preprocessing methodology. The preprocessing techniques referred to in the reviewed articles including stop words removal (*n* = 50), case folding (*n* = 31), stemming (*n* = 26), lemmatization (*n* = 16), punctuation and special character removal (*n* = 14), URL removal (*n* = 5), non-English word removing (*n* = 3), emoticon handling (*n* = 3), slang translation (*n* = 3), abbreviation replacement (*n* = 3), typo correction (*n* = 2), and acronym replacement (*n* = 1).

Requirement representations and features

The most frequently applied textual feature in the included works is the Bag-of-Word language model (*n* = 49), which is an effective and efficient method to convert text into the numerical format. Apart from the BOW model, various grammatical features are

introduced by the selected papers, including POS n-gram (Kurtanovic and Maalej, 2017*b*), the frequencies of POS (Noun/Verb/Adj/Adv/Modal) (Hussain *et al.*, 2008; Liu *et al.*, 2013; Kurtanovic and Maalej, 2017*a*), the frequency of keywords (Halim and Siahaan, 2019), and the number of syntax sub-tree (Kurtanovic and Maalej, 2017*a*, 2017*b*; Dalpiaz *et al.*, 2019).

Some statistics of sentences are applied to represent text, such as the number of characters (Abualhaija *et al.*, 2019), the number of words (Kurtanovic and Maalej, 2017*b*), the number of sentences (Qi *et al.*, 2016), the number of paragraphs (Parra *et al.*, 2015), and the number of words per sentence (Ormandjieva *et al.*, 2007).

Metadata of UGC data is applied by several works to make requirement representation more informative. A few articles have utilized metadata to provide supplementary information on requirement representations, such as the average star ratings (Maalej *et al.*, 2016) and the total number of reviews (Martens and Maalej, 2019). In addition, platforms contain metadata about users, such as the total number of reviews/ratings of the user performed (Martens and Maalej, 2019) and the platform level of the user (Qi *et al.*, 2016), are also included in the feature construction.

Domain knowledge is one of the supportive information to represent requirements. The domain knowledge is reflected by domain-specific terms, for example, the number of design terms (Parra *et al.*, 2015) and the number of keywords from the domain (Hussain *et al.*, 2008; Stanik *et al.*, 2019).

Among included works, a large proportion of recent works have used word embedding techniques to represent requirements. Word2vec (Mikolov *et al.*, 2013), FastText (Joulin *et al.*, 2017), and BERT (Devlin *et al.*, 2019) are the most widely applied embedding models across the included works (*n* = 14, 16%).

Several other features, such as the verb tense (Stanik *et al.*, 2019), the elapsed days (Liu *et al.*, 2013), the temporal tags (Abad *et al.*, 2017), the number of subjective/objective sentences in a user review (Liu *et al.*, 2013), the number of ambiguous expressions in requirements (Ormandjieva *et al.*, 2007; Parra *et al.*, 2015), the number of the sentence referring product characteristics (Liu *et al.*, 2013; Qi *et al.*, 2016), smoothed probability measure (SPM) and unsmoothed probability measure (UPM) (Hussain *et al.*, 2008), and named entities (Abad *et al.*, 2017), are also observed from the reviewed articles.

Machine learning techniques

Most of the selected works (*n* = 67) applied classification algorithms for classifying textual documents. Naïve Bayes (*n* = 33), Support

Vector Machine (SVM) ($n = 29$), Decision Tree (DT) ($n = 22$), and neural networks ($n = 26$) are the most widely applied algorithms among the studies. In addition, several specific neural network frameworks are mentioned in the included papers, which are Convolutional Neural Networks (CNNs) ($n = 12$), Feedforward Neural Networks (FNNs) ($n = 9$), and Recurrent Neural Networks (RNNs) ($n = 5$). Besides, the reviewed articles present a variety of other supervised machine learning algorithms including logistic regression ($n = 14$), K-nearest neighbors ($n = 8$), and random forest ($n = 5$).

In the included studies, two types of unsupervised algorithms were applied: text clustering techniques ($n = 4$) and topic models ($n = 13$). Clustering algorithms such as Hierarchical Agglomerative Clustering (Mahmoud, 2015; Al-Subaihin *et al.*, 2016), K-medoids (Barbosa *et al.*, 2015), and X-means (Suryadi and Kim, 2019) were utilized in the selected papers. Topic models are applied to extract main topics from the textual documents, including the Latent Dirichlet Allocation (LDA) algorithm (Blei *et al.*, 2003) ($n = 6$), aspect and sentiment unification model (Jo and Oh, 2011) ($n = 3$), Bi-term Topic Model (Yan *et al.*, 2013) ($n = 3$), and Tag Sentiment Aspect (Tang *et al.*, 2019) ($n = 1$).

Model evaluation methods

The evaluation metrics are employed differently in supervised and unsupervised approaches due to the mechanical differences between their learning methods. Manually annotated data corpus for supervised ML algorithms is applied for training and validation purposes. Hence, comparing machine predictions with actual values on a labeled dataset is a simple, straightforward way to evaluate a learning algorithm. For regression models, metrics *Mean Absolute Error* (MAE) and *Root Mean Square Error* (RMSE) are common error functions to reflect the accuracy of regression methods (Chai and Draxler, 2014). Common metrics for classification tasks are precision, recall, accuracy, f-score, and area-under-curve (AUC) (Chai and Draxler, 2014). The precision ($n = 58$), recall ($n = 56$), and F1 score ($n = 49$) are the most applied metrics to evaluate a supervised classifier by the included works. Due to the differences in data and research questions, it is difficult to compare the included works.

To evaluate the performance of the unsupervised methods, the Silhouette score was used by the included studies ($n = 4$) to assess clustering outcomes (Barbosa *et al.*, 2015; Mahmoud, 2015; Al-Subaihin *et al.*, 2016; Abad *et al.*, 2017). Furthermore, five papers applied perplexity to analyze the goodness of unsupervised categorization (Massey *et al.*, 2013; Chen *et al.*, 2014; Wang *et al.*, 2018*b*; Zhou *et al.*, 2020; Joung and Kim, 2021). Similar to the supervised method, some works manually built a golden standard to evaluate unsupervised methods with precision, recall, and F1 score (Carreno and Winbladh, 2013; Chen *et al.*, 2014; Guzman and Maalej, 2014; Abad *et al.*, 2017). In addition, several reviewed articles examine the performance with manual inspection (Massey *et al.*, 2013; Al-Subaihin *et al.*, 2016; Guzman *et al.*, 2017; Gulle *et al.*, 2020).

Tools

Most of the included works are built upon existing open-access tools and libraries. *Scikit-learn*[1] and *Waikato Environment for Knowledge Analysis*[2] *(Weka)* are the two most popular ML tools mentioned in the included articles (Hall *et al.*, 2009; Pedregosa

et al., 2011). Seventeen works applied *Scikit-learn* to build different kinds of algorithms such as Naïve Bayes, Support Vector Machine, and Random Forest. Another popular tool in the reviewed articles is the Weka, with 19 articles reporting that they applied Weka for building their solutions. Both *Scikit-learn* and *Weka* provide ready-to-use learning algorithms and have numerous tricks for preprocessing and feature extraction. For example, Wang *et al.* (2018*a*) used the *StringToWordVector* package from *Weka* to produce TF-IDF word vectors. In contrast, Dekhtyar and Fong (2017) applied *TfidfVectorizer* from the *Scikit-learn* library for the same purpose.

For natural language processing (NLP), the most popular tool is the *Natural Language Toolkit* (NLTK),[3] a Python library designed specifically for human language processing (Loper and Bird, 2002). The NLTK library is applied in selected papers for numerous preprocessing and feature extraction tasks, such as tokenization (Rahman *et al.*, 2019), sentiment analysis (Noei *et al.*, 2021), part-of-speech tagging (Halim and Siahaan, 2019), lemmatization (Guzman and Maalej, 2014), and stemming (Jha and Mahmoud, 2019). For POS and dependency parsing tasks, tools from the Stanford NLP group are mentioned, such as *Stanford parser*, *CoreNLP*, and *POS tagger*. *TensorFlow* and its high-level wrapper *Keras* are the most often used neural network libraries in the listed studies.

Findings

The purposed work is a literature review paper, which collect, review, analysis, synthesis, and report existing works based on PRISMA methodology, which aim to provide audience a summarized knowledge in ML-based requirement eliciation. The findings from the literature review will be discussed in this section. The section is organized according to the order of our research questions. In addition, the articles included in this review are categorized according to the different perspectives on the research questions. The summarization of our categorization is illustrated in Figure 3.

What requirements elicitation activities are supported by ML?

After analyzing the selected 86 papers in-depth, 15 different ML-based requirement elicitation tasks are identified (as shown in Figure 4). The identified tasks can be categorized into four main categories, which are *Preparation*, *Collection*, *Inspection*, and *Negotiation*.

Preparation refers to a set of activities that engineers must undertake before the elicitation of requirements to ensure that the process is supported by sufficient knowledge. A total of five articles are proposed to extract knowledge about the design from textual documents. For example, Liu *et al.* (2007) proposed an SVM-based design knowledge acquisition framework that can collect research articles according to organizational design knowledge taxonomy.

In addition, extracting user preferences, requests, and complaints from massive UGC is also considered a *Preparation* task. The ML-based text mining algorithms would be used to extract useful information from UGC, providing engineers with insights and knowledge about the target product. For example, Maalej *et al.* (2016) proposed a supervised method to automatically classify user app reviews into four predefined categories: user experience, bug report, feature quest, and ratings.

Liu *et al.* (2013) present a regression model which enables engineers to estimate the usefulness of customer reviews. UGC
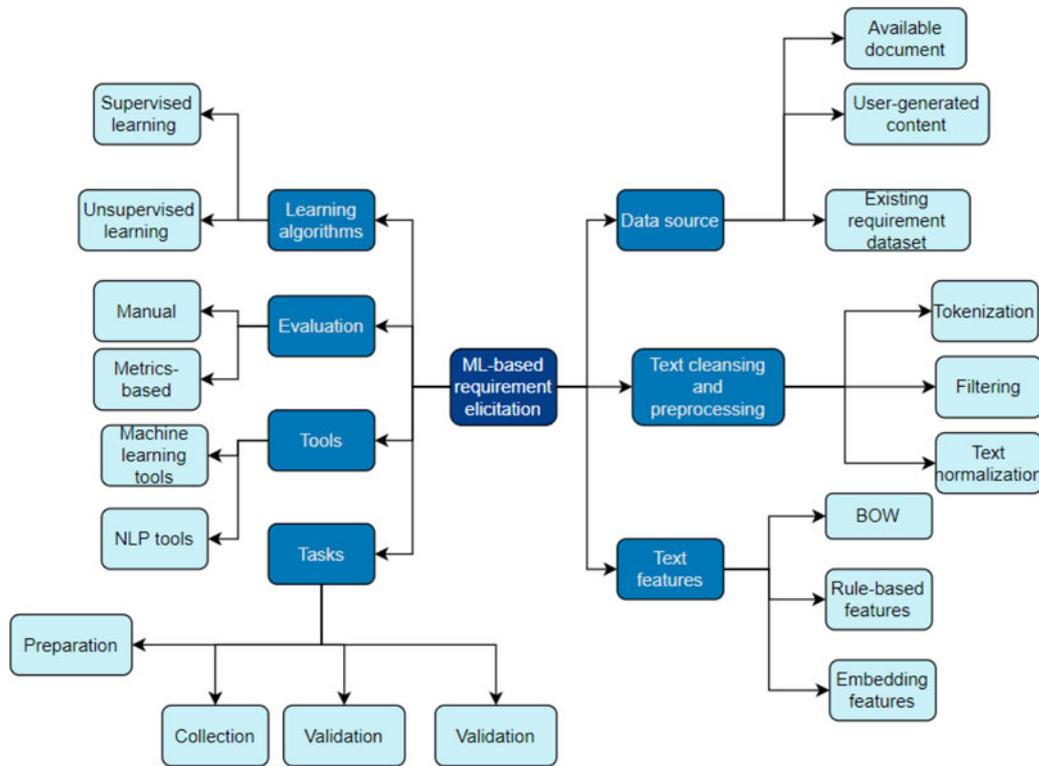
---

[1] https://scikit-learn.org/stable/index.html.
[2] https://www.cs.waikato.ac.nz/ml/weka/.

[3] https://www.nltk.org/.

Fig. 3. An illustration of the categorization schema of the collected studies.

Preparation	Collection	Inspection	Negotiation
Product knowledge acquisition (n=6, 6.98%) e.g. (Liu et al., 2007)	Requirement identification (n=6, 6.98%) e.g. (Winkler & Vogelsang, 2017)	Equivalent detection (Falessi et al., 2010)	Requirement change support (Khelifa et al., 2018)
UGC helpfulness prediction (n=3, 3.49%) e.g. (Qi et al., 2016)	Requirement classification (n=17, 19.77%) e.g. (Myllynen et al., 2021)	Fake review extraction (Martens & Maalej, 2019)	
UGC Categorization (n=11, 12.79%) e.g. (Maalej et al., 2016)	NFR/FR classification (n=15, 17.44%) e.g. (Cleland-Huang et al. 2007)	Requirement dependency extraction (Deshpande et al., 2019)	Requirement distribution (Lyutov et al., 2019)
User preference extraction (n=16, 18.60%) e.g. (Fu et al., 2013)	Security requirement identification (n=4, 16.28%) e.g. (Riaz et al., 2014)	Requirement quality assessment (Parra et al., 2015) & (Ormandjieva et al., 2007)	Refactoring decision support (Nyamawe et al., 2019)

Fig. 4. ML-based requirement elicitation tasks.

helpfulness analysis helps determine whether users' feedback is constructive. However, evaluating usefulness is a subjective activity that often entails a viewpoint. In a data-driven approach, the annotators represent the viewpoint. This review identifies two perspectives including designer-perspective (Liu *et al.*, 2013; Qi *et al.*, 2016) and consumer-perspective (Chen *et al.*, 2016).

Stakeholder preference (or tendency, rationale) is another activity categorized as *Preparation*. Since UGC is the cumulative contribution of users over some time, it incorporates their preferences and emotions about the product, product functions, and product features. For example, combining the LDA and sentiment analysis techniques can help engineers to explain which features of the product are loved by users (Guzman and Maalej, 2014; Zhou *et al.*, 2020), and which are the most dissatisfied product characteristics (Fu *et al.*, 2013).

The second group of tasks is *Collection*, which includes tasks related to directly extracting requirements or identifying specific types of requirements from a given collection of documents. In selected articles, all ML-based solutions in this category are supervised methods. The first type of collection task is requirement identification, which refers to the activity to determine whether a given sentence or paragraph is a user requirement. For example, Kengphanphanit and Muenchaisri (2020) proposed a requirement identification framework named ARESM, which can distinguish whether a given text is a requirement or non-requirement.

Requirement classification is another task in the *Collection* category. The objective of this task is to categorize the given requirements based on a certain concern. For example, Hussain *et al.* (2008) proposed a decision tree algorithm that can classify natural language requirements into functional requirements (FRs) and non-functional requirements (NFRs). The NFRs/FRs classification task takes NFRs or FRs as input and classifies them further into fine-grained subcategories. Cleland-Huang *et al.* (2007) proposed a TF-IDF-based classification algorithm that is capable of classifying textual requirements into predefined NFR subcategories. For this purpose, Cleland-Huang *et al.* (2007) established a manually labeled dataset for NFR classification.

The last type of task identified in the *Collection* is security requirement identification. Riaz *et al.* (2014) trained a K-NN classifier that can automatically detect six predefined security requirement types from natural text documents. Two articles introduce binary classifiers for identifying security requirements from written requirements (Li, 2018; Kobilica *et al.*, 2020). Jindal *et al.* (2016) trained a decision tree to further categorize security requirements into four specific categories, which are authentication-authorization, access control, cryptograph-encryption, and data integrity.

The *Inspection* and *Negotiation* could happen at any stage during a requirement engineering process. *Inspection* refers to the ML-based methods applied to inspect and assure the quality and validity of the requirements. The *Inspection* category includes equivalent requirement detection (Falessi *et al.*, 2010), requirement quality support (Ormandjieva *et al.*, 2007; Parra *et al.*, 2015), and requirement dependency analysis (Deshpande *et al.*, 2019), and fake review detection (Martens and Maalej, 2019). The *Negotiation* category includes activities to support resolving requirement-related conflicts, and there are three types of tasks were identified under this category. An SVM classifier was used by Khelifa *et al.* (2018) to automatically classify users' change requests into functional change and technical change, thereby assisting project managers to negotiate requirements and make appropriate decisions. In a recent paper, Lyutov *et al.* (2019) presented a supervised learning-enabled workflow that facilitates the automatic transmission of customer requirements to the corresponding department to facilitate the process of requirement negotiation. Moreover, an ML-based software refactoring recommendation method is proposed to assist decision-makers in deciding which major update should be applied according to customers' requests (Nyamawe *et al.*, 2019).

What data sources are used to build ML-based requirement elicitation solutions?

Based on an in-depth analysis of included studies, we found that current studies heavily rely on three types of data sources: *Textual Documents*, *UGC*, and *Existing Requirement Datasets* (Fig. 5). The category *Textual Documents* includes product requirement specification (RS) from actual projects ($n = 9$), RS from open-access online resources ($n = 8$), user stories (Barbosa *et al.*, 2015; Rodeghero *et al.*, 2017), policy documents (Massey *et al.*, 2013), and research publications (Liu *et al.*, 2007).

DePaul's NFRs corpus is the most extensively used *Existing Requirement Datasets*, which was originally introduced by Cleland-Huang *et al.* (2006). The dataset is manually labeled by graduate students from DePaul University into 10 NFR subcategories and one functional requirements category including *Availability, Look and Feel, Legal, Maintainability, Operational, Performance, Scalability, Security, Usability,* and *FRs*. In total,
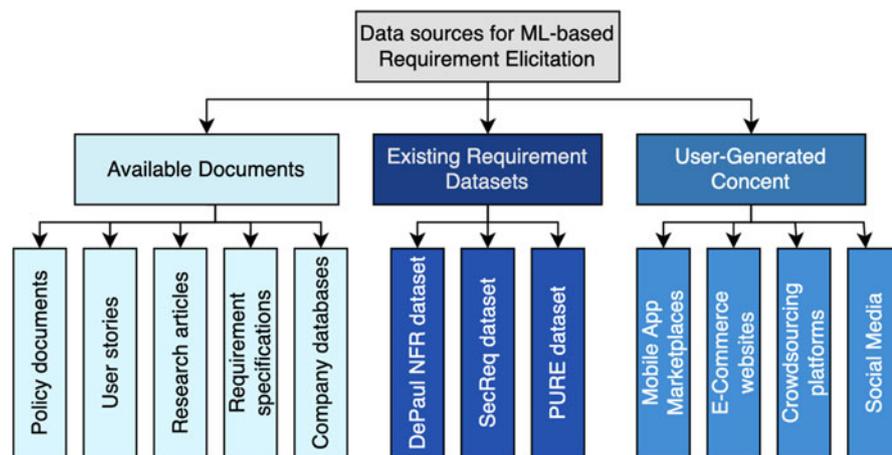

Fig. 5. The data source for building ML-based requirement elicitation solutions.

the dataset contains 358 FRs and 326 NFRs from 15 different RS. Follow-up studies applied the DePaul NFR dataset to build binary classifiers to distinguish between FR and NFR (Hussain et al., 2008; Canedo and Mendes, 2020), or multi-class classifiers to assign requirements to finer categories (Abad et al., 2017; Rahman et al., 2019).

SecReq is another publicly available requirement dataset, which was created to assist in the early stages of security requirement elicitation (Houmb et al., 2010). The dataset contains three projects, which are Electronic Purse, Customer Premises Network, and Global Platform Specification. Three projects contain 511 requirements that are tagged as security-related requirements (*sec*) and non-security-related requirements (*non-sec*). Three works trained and tested their data-driven requirement elicitation methods with SecReq corpus (Dekhtyar and Fong, 2017; Li, 2018; Kobilica et al., 2020).

The PURE dataset has 79 requirement specifications including about 35,000 sentences with an average length of 15 words (Ferrari et al., 2017). Unlike the previously described two datasets, the PURE is not labeled; rather, the authors made it open for a variety of applications. Deshpande et al. (2019) studied requirement dependencies with the PURE corpus, and EzzatiKarami and Madhavji (2021) merged both DePaul NFR and PRUE datasets for constructing a bigger training set for their study.

User-generated data (UGC) is another important source for data-driven requirements elicitation. Research shows that the needs of system users are hidden in rich UGC, such as user feedback, social networks, online software markets review, and product discussion forums (Maalej et al., 2015, 2016; Lu and Liang, 2017; Perini, 2018). UGC contains any form of data generated by users, like numerical ratings, textual product reviews, and videos. In total, half of the included studies (*n* = 43) applied UGC to build their ML-based solutions. The UGC source includes mobile application platform user reviews (Apple App Store and Google Play Store), e-commerce user reviews (Amazon and other online retailers), social media (Twitter and Facebook), and crowdsourcing platforms.

What technologies, algorithms, and tools are used to build ML-based requirement elicitation?

This subsection answers RQ3 and RQ4. Our study identified the technical approaches and algorithms used by the included studies and divided them into three categories: *Textual Data Cleansing and Preprocessing, Textual Features Extraction,* and *Machine Learning* (ML) (Fig. 6). The ML models are evaluated by two strategies, which are *Manual evaluation* and *Metrics-based evaluation*. In addition, we categorized many open-source tools identified from the reviewed articles into two categories: *ML tools* and *NLP tools*.

Textual data cleansing and preprocessing

Twenty different techniques were identified from the included papers specifically for cleaning and preparing data, which we categorized under the *Textual Data Cleansing and Preprocessing* category. In addition, due to the functional features of these
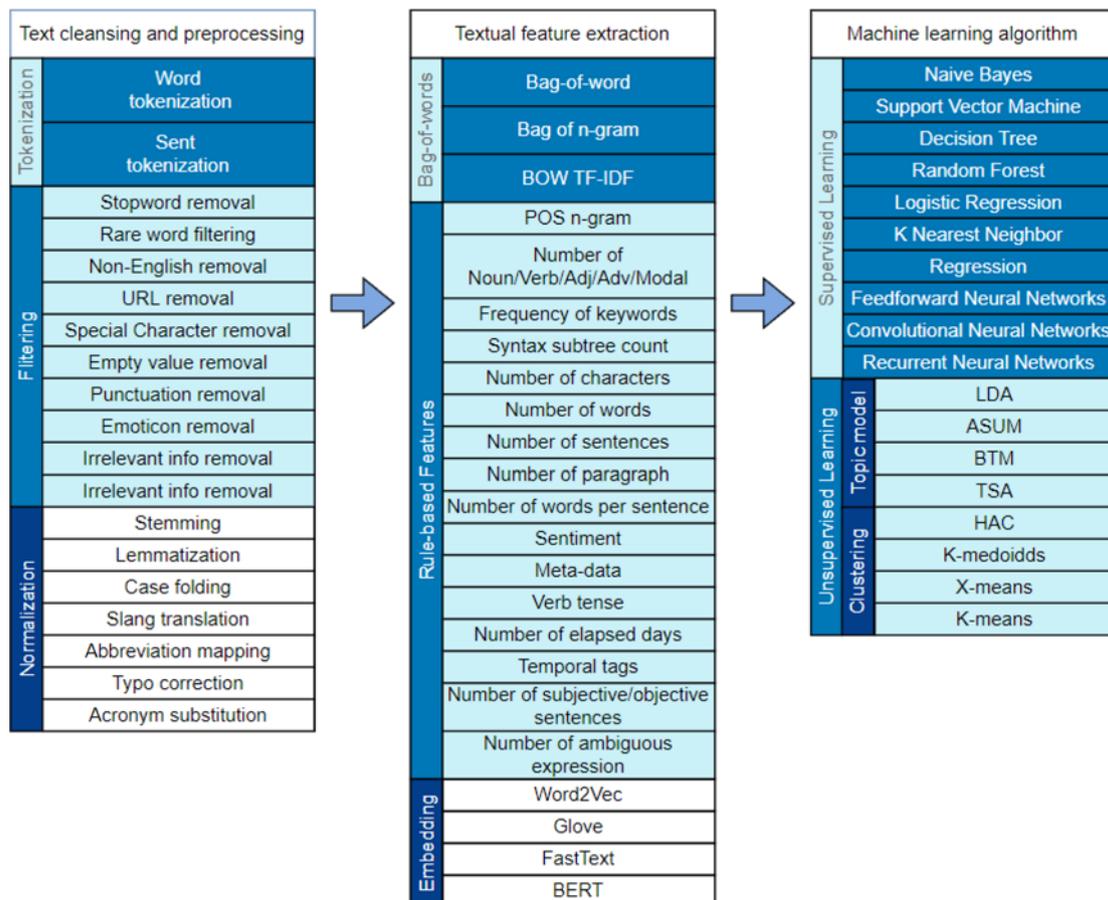

Fig. 6. Technologies and algorithms.

techniques, we further grouped these techniques into three parts: *tokenization*, *text chunking*, and *text normalization*.

Tokenization is a procedure to break a given sequence of text down into smaller parts, such as breaking a document into sentences (sentence tokenization) or breaking a sentence into individual words (word tokenization). *Text filtering* is a group of preprocessing methods, which aim to eliminate redundant, erroneous, non-representative, inconsistent, and ineligible data from a text document. In the reviewed articles techniques include stopwords removal, rare word filtering, non-English word removing, URL removing, special character handling, empty value handling, punctuation removal, emoticon handling, non-informative/irrelevant word removing, and inconsistent information removal are considered under this classification. *Text normalization* aims to transform a text sequence into a standard form to reduce its randomness. Stemming and lemmatization are the most common text normalization methods. In a document, a word has various forms, and some of these forms can be converted to one another by adding or removing the prefix or suffix (Manning *et al.*, 2008). Stemming is a crude heuristic procedure that removes the tails from words to get word stems, which are the fundamental word units, such as for word requirements, the word stem is required (Manning *et al.*, 2008). In comparison, lemmatization yields a basic dictionary form of a word. For example, the lemmatization of requirements will yield requirements. Case folding is another popular text normalization approach that changes all letters in a word into lower cases (Manning *et al.*, 2008). In addition, slang translations, abbreviation translations, typo corrections, and acronym substitutes are considered text normalization procedures since they convert text into a more generic form.

Textual features extraction

Textual Features Extraction includes a set of techniques to convert natural text into numbers. We found three major textual data representation strategies from the reviewed articles: *Bag-of-word*, *Rule-based*, and *Embedding* features. The Bag-of-word considers a sequence of text as a set (or multi-set) of the word regardless of word order and grammar (Manning and Schütze, 1999). Various BOW representation strategies can be found in the included works, such as using simple raw counts for words, a bag of bigram or trigram (Kurtanovic and Maalej, 2017*a*), and BOW with TF-IDF weighting (Li *et al.*, 2018).

In addition to BOW features, studies included in this review also applied rule-based handcraft features, such as POS n-gram (Kurtanovic and Maalej, 2017*b*), the number of Noun/Verb/Adj/Adv/Modal (Hussain *et al.*, 2008; Liu *et al.*, 2013; Kurtanovic and Maalej, 2017*a*), frequency of POS of the keywords (Halim and Siahaan, 2019), and the count of syntax sub-tree (Kurtanovic and Maalej, 2017*a*, 2017*b*; Dalpiaz *et al.*, 2019). In addition, textual descriptive statistics are also applied to represent requirements, including the number of characters (Abualhaija *et al.*, 2019), word count (Kurtanovic and Maalej, 2017*b*), sentence count (Qi *et al.*, 2016), paragraphs count (Parra *et al.*, 2015), and the number of words per sentence (Ormandjieva *et al.*, 2007). Furthermore, temporal features including verb tense (Stanik *et al.*, 2019), the number of elapsed days (Liu *et al.*, 2013), and temporal tags, such as time, duration, and time set (Abad *et al.*, 2017), were used to represent the temporal information of the requirements. For UGC-based research, some platforms provide metadata that can be extracted to represent user comments. Metadata features include star ratings (Maalej *et al.*, 2016), review count (Martens and Maalej, 2019), and the number of links (Parra *et al.*, 2015).

Moreover, some studies applied document quality features to represent textual requirements, including the number of subjective/objective sentences in a review (Liu *et al.*, 2013), the number of ambiguous expressions in a requirement (Ormandjieva *et al.*, 2007; Parra *et al.*, 2015), and the number of the sentence referring product feature appeared in a user review (Liu *et al.*, 2013; Qi *et al.*, 2016). Additionally, some articles introduce domain-specific features, such as the number of design terms (Parra *et al.*, 2015) and the number of keywords from the input text (Hussain *et al.*, 2008; Stanik *et al.*, 2019).

In recent years, word embedding has gained popularity in a range of NLP applications. The selected articles used a range of embedding techniques, including Word2vec (Mikolov *et al.*, 2013), FastText (Joulin *et al.*, 2017), Glove (Pennington *et al.*, 2014), and BERT (Devlin *et al.*, 2019) to represent words. Three strategies associated with embedding features are identified in the included studies: training the embedding from scratch using a pre-trained embedding and fine-tuning the previously trained language models.

Machine learning

In this review, the learning algorithms applied by the included studies are categorized into two categories: *supervised* and *unsupervised learning*. Under *supervised learning* categories, only three studies applied regression models (Liu *et al.*, 2013; Chen *et al.*, 2016; Qi *et al.*, 2016). The regression methods can help engineers to predict a numerical value to reflect the helpfulness of a given user review. The rest of the methods in *supervised learning* are all classification algorithms. *Topic modeling* and *clustering* techniques are two frequently applied *Unsupervised Learning* methods and the LDA is the most widely applied unsupervised method in the papers included.

Evaluation methods

The quality of models can be reflected in the evaluation metrics, which are a set of formulas and units of measurement that reflect how well the learning algorithm could perform (Hossin and Sulaiman, 2015). For different types of learning tasks, the evaluation methods are used differently. In the included studies, the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are employed for regression models. Both MAE and RMSE are negatively oriented, which means the better the model, the lower the errors. Precision, Recall, and F1 score are most frequently applied for classification models. On the other hand, the unsupervised method is evaluated by two strategies: internal and external evaluation. The included works applied intra and inter-cluster similarity (or distance), Silhouette score, and perplexity to assess the clustering results for internal evaluation. In the case of external evaluation, domain experts are asked to evaluate the models' results manually. Additionally, a truth set can be built to evaluate the clustering results, similar to a supervised classifier.

Available tools

The included studies widely mentioned two types of tools: ML tools and NLP tools. The NLP tools such as NLTK and CoreNLP are applied to preprocess and extract the features from the textual data. The most widely mentioned ML tools are Weka and Scikit-learn, which integrate multiple ML algorithms and quickly build a data-driven solution. Keras is a popular deep learning library among the included studies, which contains the most popular neural network architectures with compact and straightforward APIs.

Table 5. Tools mentioned by included works

Category	Sub-categories	Tool name
ML tools	*ML*	WEKA, Scikit-learn, MATLAB, Classifier4J (Java)
	Deep learning	TensorFlow, Keras
	Language/topic model tools	JLDADMM (Java), Stanford topic model toolbox (Java), Spacy (Python), Genism (Python), tm text mining package (R)
NLP tools	*Multi-purpose NLP tools*	NLTK (Python), LingPipe (Java), koRups package (R), CoreNLP (Java), Stanford NLP toolkit (Java/Python), WordNet, koRpus statistical readability package, IBM Watson NLU
	Sentiment analysis tools	SEMAFOR (Java), SentiStrength (Java), VaderSentiment (Python), Textblob
	Parsing tools	MaltParser (Java), Stanford POS tagger, Stanford parser, Berkeley parser, Stanford temporal tagger
	Single-purpose tools	Jazzy (spell-checker), Jieba (Chinese character tokenizer)

Table 5 lists the tools mentioned in the reviewed articles, arranged by their uses. In total, seven types of tools are extracted: ML tools (conventional), deep learning tools, language, and topic model tools, multi-purpose NLP tools, sentiment analysis tools, parsing tools, and single-purpose tools.

How to construct an ML-based requirements elicitation method?

According to the included studies, building an ML-based requirements elicitation method contains four major steps: study design, data preparation, model construction, and model implementation. The first step is to design the ML-based requirements elicitation study by considering two fundamental elements: identifying the requirements elicitation subtasks and available datasets. In this literature review, we identify four major tasks and three types of data sources that support requirements elicitation from the reviewed articles.

Different ML-based requirement elicitation tasks require different datasets and data annotation strategies. Hence, studies should be designed differently accordingly. Both requirement documents and UGC data were applied in the selected research. Through the detailed review of the included papers, we identified that the requirements are usually stored in plain text format, and the corresponding tasks are mostly focusing on requirement text classification. However, the UGC data contains additional metadata that describes the data in many aspects, such as ratings and timestamps. With these additional data, researchers can design studies such as the prediction of usefulness associated with user reviews (Liu *et al.*, 2013), and analysis of user preferences on a timely basis (Fu *et al.*, 2013).

In addition, sentiment is another reason that causes differences in requirement document analysis and UGC analysis. Documents describing the requirements are usually written with neutral language; therefore, analyzing the sentiment of each requirement may not be as significant as analyzing the sentiment of UGC. As a result, sentiment analysis does not appear in the requirement document analysis tasks but is commonly used in UGC-based research.

With the defined task and dataset, the next step is to construct an ML pipeline, which relies heavily on the understanding of machine learning and NLP techniques. Data cleansing, data preprocessing, feature extraction, model training, and model evaluation are part of this phase. Though the model construction pipeline can be independent of domain knowledge for unsupervised learning, domain expertise is still necessary to validate and evaluate the models. Finally, the model implementation is an important final step to build an ML-based requirements elicitation. Multiple aspects must be considered, such as organizational culture, management, security, development, and operation procedure.

Open issues and future works

It is important to note that eliciting requirements is not one single activity, rather it comprises multiple sessions and operations that work together as a whole. However, there is no very detailed definition or uniform approach to this stage in academia and industry. For example, Young (2004) suggested a twenty-eight-step requirement gathering activities checklist including planning, managing, collecting, reviewing, tracing, etc. Wiegers and Beatty (2013) summarizes 21 best practices for requirements elicitation, including defining scope, identifying stakeholders, reusing existing requirements, modeling the application environment, and analyzing requirements feasibility. Using a single ML model cannot accomplish so many different tasks. Therefore, ML techniques are only able to accomplish partial tasks involved in requirement elicitation. Furthermore, most of the included studies are all focusing on resolving a particular task with ML, rather than designing a complete system that supports requirement elicitation. In this regard, most of the ML-based methods developed so far have a supporting or complementary role to traditional methods. For example, in an ML-aided requirement elicitation system, conventional methods, such as interviews, questionnaires, and brainstorming, are responsible for producing and collecting requirement-related data. ML algorithms, however, are responsible for analyzing data or supporting follow-up data-related activities.

In Section "What requirements elicitation activities are supported by ML?", we summarized 15 ML-based requirements elicitation subtasks from included studies and categorize them into four groups. Most works were classified as *Preparation* ($n = 37$) and *Collection* ($n = 41$) tasks, and only eight articles were identified as *Validation*

($n = 5$) and *Negotiation* ($n = 3$) tasks. One reason for this is that the validation and negotiation are hard to articulate due to the high complexity of the tasks. For example, tasks from *Negotiations* require collaboration, discussion, and trade-offs between stakeholders from many aspects. Therefore, most of the challenges related to these tasks are related to background knowledge, communication, budgets, or other limitations imposed by the real world. As a result, it is difficult to model these tasks correctly.

It is still challenging to build an ML-based solution to fully automate requirement elicitation. First, since requirement elicitation is a comprehensive process composed of a variety of tasks and goals, it is difficult to develop an end-to-end ML model to fully automate the requirement elicitation process. Second, requirements could come from a large variety of sources, particularly in the big data era. In terms of data type and format, the datasets included in the study were highly heterogeneous. For example, sentiment analysis may be useful when analyzing UGC data, but it is not valuable when analyzing neutral document data. Hence, using the model specifically designed for UGC, such as ASUM (Jo and Oh, 2011), cannot perform as expected on document data, and vice versa. Third, the ML-based requirement elicitation approach is automatic but easily affected by errors and failure. Unlike rule-based systems that can be debugged and fixed locally in the coded knowledge body, it is difficult to directly tune the ML model when dealing with known errors. In addition, the interpretation of ML models is still an open challenge in academia and industry. For example, deep neural networks learn features automatically, which makes it more challenging to analyze the reasons behind ML-based solutions. Furthermore, only a few research considered the changing nature of the requirements. Due to the dynamic nature of the requirements, in practice, requirement elicitation requires engineers to identify and modify requirements based on the unpredictable nature of user needs (Xie *et al.*, 2017). Besides, in terms of both content and type of task, the current research is monotonous. The vast majority of studies still focus on classification and clustering.

To tackle these challenges, the following future research directions are suggested by the authors. First, although there are growing interests and works in building ML-based requirement elicitation methods, there is still a vacancy for a systematic guide on how to integrate the ML-based components into the requirement elicitation framework. Multiple aspects of the integrated system should be considered, such as how humans and machines interact in requirements elicitation, what is the input–output of the system and each subsystem, and what specific tasks should be performed by machines when expert involvement is required, among others. Hence, a systematic study and guidance of AI system design, engineering, implementation, and management are required.

Second, there is a lack of in-depth analysis of ML-based requirement elicitation failure and errors. For example, research papers and projects typically rely on statistical metrics for ML model validation and evaluation. This type of evaluation can tell us how good or bad a model is, but neglects to address the question of what leads a model to perform unexpectedly. Future studies should address this issue by introducing methods and techniques to explore the factors that affect the performance of ML-based requirement elicitation.

Third, the ML-based methods, especially deep learning models are lacking transparency. Because deep neural networks derive their features not from experience and knowledge, but from data, which is more effective but less intuitive. Since requirement elicitation is knowledge-intensive human-involved activity, the engineers not only expect models to solve the problems but also to explain them. The significance of Explainable AI (XAI) is increased along with the widespread adoption of deep learning methods in recent years (Xu *et al.*, 2019). In the future, research in ML-based engineering of requirements will also need to leverage XAI techniques and methods to investigate the nature of decision-related requirements.

Forth, a broad range of NLP tasks could be incorporated into the requirements elicitation. Apart from text classification, many other NLP techniques can be utilized to support requirements elicitation, such as neural summarization, text generation, neural conversational bots, question asking, question answering, and text to speech. Due to its wide range of tasks, requirements elicitation provides an excellent opportunity to practice cutting-edge NLP methods. Future research works should try more to incorporate these methods into requirement elicitation. As an example, neural text generation technologies such as Seq2Seq (Sutskever *et al.*, 2014), GAN (Goodfellow *et al.*, 2014), and T5 Text-to-Text transformers (Matena *et al.*, 2019) have the potential to produce new mock requirements based on a particular context, which may provide innovative data-driven ideas from a new perspective.

Fifth, aside from natural text, user needs also can be mined from other data formats. E-commerce platforms, for instance, allow individuals to upload videos and pictures to share usage experiences, complaints, and feedback. Although techniques such as neural image description (Vinyals *et al.*, 2015; Karpathy and Li, 2017) and neural video description (Yao *et al.*, 2015) are not as mature as text classification techniques, they are also of great research value and can play a major role in requirement engineering as well.

Sixth, due to the data-intensive nature of ML methods, more requirements related to high-quality text data should also be introduced. However, some interest-related requirements are requested to be kept confidential by the relevant stakeholders. Hence, sharing high-quality requirement data with the requirement engineering community is challenging. Masking sensitive data or substituting entities can be effective means of modifying sensitive requirements, which can facilitate the sharing of information within the requirement engineering community. Another strategy to address insufficient training data is to develop a language model specifically for requirements engineering. Research shows that transfer learning techniques can overcome the limitations of insufficient data (Howard and Ruder, 2018). Future works could also consider building neural language models that are specifically trained with requirement specifications.

Seventh, since user-requirement elicitation is a human-centric activity, analyzing user behavior may provide valuable insight into understanding and eliciting requirements. As the study of representation learning, such as user embedding, is being applied to a variety of different domains, including recommendation and healthcare systems (Miotto *et al.*, 2016; Pan and Ding, 2019). Analyzing user behavior can help to predict user preference and explore potential requirements change.

Last, future work should address the issues caused by the dynamic nature of user requirements. In practice, stakeholder requirements are not always static; however, in the studies reviewed, ML algorithms were used to read the static text to identify requirements. Further research on ML-based methods should be focusing on changing requirements and reducing their impact are urgently needed.

Limitation of this review

We used PRISMA as the research framework to identify the primary research studies in this review. Unlike other popular methods, such

as snowballing approach, in this study, we did not exhaustively identify further relevant studies by iterating through the reference lists. This review chose to use minimum evidence to reflect the current state of ML-based requirements elicitation rather than providing an exhaustive result. Thus, some relevant studies may have been omitted from this review. In addition, there is a paradox between literature review and search query generation. Before a literature review is completed, it is not easy to define a set of exact keywords to represent the topic. Simultaneously, the absence of good search queries and keywords could defy the effort to retrieve relevant papers effectively. Hence, it is challenging to develop a perfect set of search queries at the initial stage that covers all of the aspects related to the field. To deal with these issues, we dynamically adjusted the search queries for seven academic databases to reduce bias and loss in the search results.

Numerous publications are excluded due to a lack of technical details; this does not imply that those articles are unimportant to this field. Various ideas and concepts may still be derived from these works. Moreover, only one of the similar works by the same author has been retained in the study; however, it is difficult to define a clear boundary to decide which work to keep. As a precaution to minimize the risks associated with inclusion–exclusion criteria, the authors discussed and evaluated the articles through meetings in cases wherever it was challenging to decide individually.

Additionally, human errors could not be avoided in the data extraction phase due to its nature of subjectivity. As data extraction table in Table 4 illustrated, the reviewer needs to enter two types of data manually. The first type of data is the descriptive data, which can be accessed from the academic research databases and the websites of journals. However, the second type of data requires reviewers to assess and extract information based on personal understanding. Therefore, the data extraction process inevitably contains a certain amount of bias and subjectivity. In addition, since the requirement elicitation is an interdisciplinary problem, many definitions are disputed. For example, the definition of the requirement and requirement elicitation are all defined differently by various researchers. Besides, some information was not explicitly stated in the reviewed articles, which led to difficulties in corresponding information retrieval. To overcome this limitation, the author team iterated and adjusted the data extraction table before reaching a final agreement.

Conclusion

The review provides an overview of the current research on ML-based requirements elicitation. First, we categorized the included studies into four ML-based requirement elicitation tasks: *Preparation*, *Collection*, *Validation,* and *Negotiation*. Second, we examined the data sources and corpora used by the included studies to develop the machine learning models for requirements elicitation. As a result, we identified three types of data sources for building ML solutions, which are *Textual Documents*, *UGC*, and *Existing Requirement Datasets*. Third, in this review, general ML pipelines are extracted from the included studies: text cleansing and preprocessing, textual feature extraction, machine learning, and evaluation. Furthermore, we identified 19 tasks among the selected works and assigned them to three types of text cleaning and preprocessing groups: filtering, normalizing, and tokenizing. For the text feature extraction part, we classified the included works into three groups according to the technique used to extract the features. BOW language models and handcrafted features are frequently found in reviewed

publications, but in recent years, an increasing trend towards using embedding features has been observed. In addition, we discovered the most popular algorithms, such as Naive Bayes, Support Vector Machines, Decision Trees, and Neural Networks in this review. Precision, Recall, and F1 score are the most prevalent evaluation metrics applied to assess model performance. Finally, we listed the most popular NLP tools, which are NLTK and CoreNLP, and the most commonly applied machine learning tools, Weka and Scikit-learn.

Apart from the main findings, one major observation is that most research focuses on requirements categorization tasks. There is a notable majority of papers in the collection that are focused on supervised text classification, followed by topic modeling and clustering techniques. Second, we noticed that the existing articles are more focused on using machine learning to solve specific and fine-grained problems in requirements elicitation, such as classifying NFRs and extracting main topics from massive user reviews. It has, however, been relatively rare for research to examine how to integrate machine learning-based requirements acquisition methods into existing requirements elicitation workflows. Hence, the lack of expertise in designing, engineering, implementing, and configuring ML-based requirement elicitation systems calls for further research. Furthermore, most studies lack concrete evidence that machine learning can assist designers and engineers in reducing time and effort in requirement extraction. Last, although supervised learning is prevalent in this field, we have found only two publicly accessible labeled datasets from the 86 reviewed papers: DePaul's NFRs dataset (Cleland-Huang *et al.*, 2006) and SecReq (Knauss *et al.*, 2011).

Thus far, ML-based solutions have been monolithic in eliciting requirements; however, the publications in this field provide sufficient evidence that ML can support requirements activities both theoretically and practically. A number of labor-intensive, error-prone activities from requirement engineering are waiting to be supported by ML. Despite what has already been accomplished, the best is yet to come.

References

Abad ZSH, Karras O, Ghazi P, Glinz M, Ruhe G and Schneider K (2017) What works better? A study of classifying requirements. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, Vol. **1**, pp. 496–501. doi:10.1109/RE.2017.36

Abualhaija S, Arora C, Sabetzadeh M, Briand LC and Vaz E (2019) A machine learning-based approach for demarcating requirements in textual specifications. Proceedings of the IEEE International Conference on Requirements Engineering, 2019 September, pp. 51–62. doi:10.1109/RE.2019.00017

Ahmad A, Feng C, Khan M, Khan A, Ullah A, Nazir S and Tahir A (2020) A systematic literature review on using machine learning algorithms for software requirements identification on stack overflow. *Security and Communication Networks* 2020. doi:10.1155/2020/8830683

Al-Subaihin AA, Sarro F, Black S, Capra L, Harman M, Jia Y and Zhang Y (2016) Clustering mobile apps based on mined textual features. International Symposium on Empirical Software Engineering and Measurement, 08–09 September. doi:10.1145/2961111.2962600

Asif M, Ali I, Malik MSA, Chaudary MH, Tayyaba S and Mahmood MT (2019) Annotation of software requirements specification (SRS), extractions of nonfunctional requirements, and measurement of their tradeoff. *IEEE Access* 7, 36164–36176. doi:10.1109/ACCESS.2019.2903133

Baker C, Deng L, Chakraborty S and Dehlinger J (2019) Automatic multi-class non-functional software requirements classification using neural networks. *Proceedings - International Computer Software and Applications Conference* **2**, 610–615. doi:10.1109/COMPSAC.2019.10275

Bakiu E and Guzman E (2017) Which feature is unusable? Detecting usability and user experience issues from user reviews. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017, pp. 182–187. doi:10.1109/REW.2017.76

Barbosa R, Januario D, Silva AE, Moraes R and Martins P (2015) An approach to clustering and sequencing of textual requirements. Proceedings - 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, DSN-W 2015, pp. 39–44. doi:10.1109/DSN-W.2015.20

Binkhonain M and Zhao L (2019) A review of machine learning algorithms for identification and classification of non-functional requirements. *Expert Systems with Applications: X*, **1**. doi:10.1016/j.eswax.2019.100001

Blei DM, Ng AY and Jordan MI (2003) Latent Dirichlet allocation. *Journal of Machine Learning Research* **3**, 993–1022. doi:10.1016/b978-0-12-411519-4.00006-9

Bourque P and Fairley RE (2014) *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0).* IEEE Computer Society.

Canedo ED and Mendes BC (2020) Software requirements classification using machine learning algorithms. *Entropy* **22**. doi:10.3390/E22091057

Carreno LVG and Winbladh K (2013) Analysis of user comments: an approach for software requirements evolution. Proceedings - International Conference on Software Engineering, pp. 582–591. doi:10.1109/ICSE.2013.6606604

Casamayor A, Godoy D and Campo M (2009) Semi-supervised classification of non-functional requirements: an empirical analysis. *Inteligencia Artificial* **13**, 35–45. doi:10.4114/ia.v13i44.1044

Casamayor A, Godoy D and Campo M (2010) Identification of non-functional requirements in textual specifications: a semi-supervised learning approach. *Information and Software Technology* **52**, 436–445. doi:10.1016/j.infsof.2009.10.010

Casamayor A, Godoy D and Campo M (2012) Knowledge-based systems functional grouping of natural language requirements for assistance in architectural software design. *Knowledge-Based Systems* **30**, 78–86. doi:10.1016/j.knosys.2011.12.009

Chai T and Draxler RR (2014) Root mean square error (RMSE) or mean absolute error (MAE)?: Arguments against avoiding RMSE in the literature. *Geoscientific Model Development* **7**, 1247–1250. doi:10.5194/gmd-7-1247-2014

Chen N, Lin J, Hoi SCH, Xiao X and Zhang B (2014) AR-miner: mining informative reviews for developers from mobile app marketplace. *Proceedings - International Conference on Software Engineering* **1**, 767–778. doi:10.1145/2568225.2568263

Chen J, Zhang C and Niu Z (2016) Identifying helpful online reviews with word embedding features. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9983 LNAI, pp. 123–133. doi:10.1007/978-3-319-47650-6_10

Cleland-Huang J, Settimi R, Zou X and Sole P (2006) The detection and classification of non-functional requirements with application to early aspects. Proceedings of the IEEE International Conference on Requirements Engineering, pp. 36–45. doi:10.1109/RE.2006.65

Cleland-Huang J, Settimi R, Zou X and Solc P (2007) Automated classification of non-functional requirements. *Requirements Engineering* **12**, 103–120. doi:10.1007/s00766-007-0045-1

Cooper HM (1988) Organizing knowledge syntheses: a taxonomy of literature reviews. *Knowledge in Society* **1**, 104–126. doi:10.1007/BF03177550

Cooper HM, Hedges LV and Valentine JC (2019) *Handbook of Research Synthesis and Meta-Analysis.* New York: Russell Sage Foundation.

Dalpiaz F, Dell'Anna D, Aydemir FB and Çevikol S (2019) Requirements classification with interpretable machine learning and dependency parsing. Proceedings of the IEEE International Conference on Requirements Engineering, 2019 September, pp. 142–152. doi:10.1109/RE.2019.00025

Dekhtyar A and Fong V (2017) RE data challenge: requirements identification with Word2Vec and TensorFlow. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, pp. 484–489. doi:10.1109/RE.2017.26

Deshpande G, Arora C and Ruhe G (2019) Data-driven elicitation and optimization of dependencies between requirements. Proceedings of the IEEE International Conference on Requirements Engineering, 2019 September, pp. 416–421. doi:10.1109/RE.2019.00055

Devlin J, Chang MW, Lee K and Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, 1(Mlm), pp. 4171–4186.

Dhinakaran VT, Pulle R, Ajmeri N and Murukannaiah PK (2018) App review analysis via active learning: reducing supervision effort without compromising classification accuracy. Proceedings - 2018 IEEE 26th International Requirements Engineering Conference, RE 2018, pp. 170–181. doi:10.1109/RE.2018.00026

El Dehaibi N, Goodman ND and MacDonald FE (2019) Extracting customer perceptions of product sustainability from online reviews. *Journal of Mechanical Design, Transactions of the ASME* **141**. doi:10.1115/1.4044522

EzzatiKarami M and Madhavji NH (2021) Automatically classifying non-functional requirements with feature extraction and supervised machine learning techniques: a research preview. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. **12685** LNCS. Springer International Publishing. doi: 10.1007/978-3-030-73128-1_5

Falessi D, Cantone G and Canfora G (2010) A comprehensive characterization of NLP techniques for identifying equivalent requirements. ESEM 2010 - Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. doi:10.1145/1852786.1852810

Ferrari A, Spagnolo GO and Gnesi S (2017) PURE: a dataset of public requirements documents. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, pp. 502–505. doi:10.1109/RE.2017.29

Fu B, Lin J, Liy L, Faloutsos C, Hong J and Sadeh N (2013) Why people hate your App - making sense of user feedback in a mobile app store. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Part F1288, pp. 1276–1284. doi:10.1145/2487575.2488202

Gnanasekaran RK, Chakraborty S, Dehlinger J and Deng L (2021) Using recurrent neural networks for classification of natural language-based non-functional requirements. CEUR Workshop Proceedings, p. 2857.

Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y (2014) Generative adversarial nets. *Advances in Neural Information Processing Systems* **27**, 3063–3071. doi:10.1109/ICCVW.2019.00369

Gulle KJ, Ford N, Ebel P, Brokhausen F and Vogelsang A (2020) Topic modeling on user stories using word mover's distance. Proceedings - 7th International Workshop on Artificial Intelligence and Requirements Engineering, AIRE 2020, pp. 52–60. doi:10.1109/AIRE51212.2020.00015

Guzman E and Maalej W (2014) How do users like this feature? A fine grained sentiment analysis of App reviews. Proceedings - 2014 IEEE 22nd International Requirements Engineering Conference, RE 2014, pp. 153–162. doi:10.1109/RE.2014.6912257

Guzman E, Ibrahim M and Glinz M (2017) A little bird told me: mining tweets for requirements and software evolution. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, Vol. **3**, pp. 11–20. doi:10.1109/RE.2017.88

Halim F and Siahaan D (2019) Detecting non-atomic requirements in software requirements specifications using classification methods. 2019 1st International Conference on Cybernetics and Intelligent System, ICORIS 2019, August, pp. 269–273. doi:10.1109/ICORIS.2019.8874888

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P and Witten IH (2009) The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter* **11**, 10–18.

Han Y and Moghaddam M (2021) Eliciting attribute-level user needs from online reviews with deep language models and information extraction. *Journal of Mechanical Design, Transactions of the ASME* **143**. doi:10.1115/1.4048819

Haque MA, Rahman MA and Siddik MS (2019) Non-functional requirements classification with feature extraction and machine learning: an

empirical study. 1st International Conference on Advances in Science, Engineering and Robotics Technology 2019, ICASERT 2019, 2019 (Icasert). doi:10.1109/ICASERT.2019.8934499

Hossin M and Sulaiman MN (2015) A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*. doi:10.5121/ijdkp.2015.5201

Houmb SH, Islam S, Knauss E, Jürjens J and Schneider K (2010) Eliciting security requirements and tracing them to design: an integration of common criteria, heuristics, and UMLsec. *Requirements Engineering* 15, 63–93. doi:10.1007/s00766-009-0093-9

Howard J and Ruder S (2018) Universal language model fine-tuning for text classification. ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers), Vol. 1, pp. 328–339. doi:10.18653/v1/p18-1031

Huang J, Gheorghe A, Handley H, Pazos P, Pinto A, Kovacic S, Collins A, Keating C, Sousa-Poza A, Rabadi G, Unal R, Cotter T, Landaeta R and Daniels C (2020) Towards digital engineering: the advent of digital systems engineering. *International Journal of System of Systems Engineering* 10, 234–261. doi:10.1504/IJSSE.2020.109737

Hussain I, Kosseim L and Ormandjieva O (2008) Using linguistic knowledge to classify non-functional requirements in SRS documents. Lecture Notes in Computer Science: Vol. 5039 LNCS, pp. 287–298. doi:10.1007/978-3-540-69858-6_28

Jeon K, Lee G and Jeong HD (2021) Classification of the requirement sentences of the US DOT standard specification using deep learning algorithms. *Lecture Notes in Civil Engineering* 98, 89–97. doi:10.1007/978-3-030-51295-8_8

Jha N and Mahmoud A (2019) Mining non-functional requirements from App store reviews. *Empirical Software Engineering* 24, 3659–3695. doi:10.1007/s10664-019-09716-7

Jindal R, Malhotra R and Jain A (2016) Automated classification of security requirements. 2016 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2016, pp. 2027–2033. doi:10.1109/ICACCI.2016.7732349

Jo Y and Oh A (2011) Aspect and sentiment unification model for online review analysis. Proceedings of the 4th ACM International Conference on Web Search and Data Mining, WSDM 2011, pp. 815–824. doi:10.1145/1935826.1935932

Jones IA and Kim KY (2015) Systematic service product requirement analysis with online customer review data. *Journal of Integrated Design and Process Science* 19, 25–48. doi:10.3233/jid-2015-0011

Joulin A, Grave E, Bojanowski P and Mikolov T (2017) Bag of tricks for efficient text classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Vol. 2, pp. 427–431. doi:10.18653/v1/e17-2068

Joung J and Kim HM (2021) Automated keyword filtering in latent dirichlet allocation for identifying product attributes from online reviews. *Journal of Mechanical Design, Transactions of the ASME* 143, 1–6. doi:10.1115/1.4048960

Karpathy A and Li F-F (2017) Deep visual-semantic alignments for generating image descriptions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3128–3137. doi:10.1017/9781316569290.011

Kengphanphanit N and Muenchaisri P (2020) Automatic requirements elicitation from social media (ARESM). PervasiveHealth: Pervasive Computing Technologies for Healthcare, pp. 57–62. doi:10.1145/3418994.3419004

Khelifa A, Haoues M and Sellami A (2018) Towards a software requirements change classification using support vector machine. *CEUR Workshop Proceedings* 2279, 1–10.

Knauss E, Houmb S, Schneider K, Islam S and Jürjens J (2011) Supporting requirements engineers in recognising security issues. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6606 LNCS, pp. 4–18. doi:10.1007/978-3-642-19858-8_2

Ko Y, Park S, Seo J and Choi S (2007) Using classification techniques for informal requirements in the requirements analysis-supporting system. *Information and Software Technology* 49, 1128–1140. doi:10.1016/j.infsof.2006.11.007

Kobilica A, Ayub M and Hassine J (2020) Automated identification of security requirements: a machine learning approach. PervasiveHealth: Pervasive Computing Technologies for Healthcare 1, 475–480. doi: 10.1145/3383219.3383288

Kurtanovic Z and Maalej W (2017a) Automatically classifying functional and non-functional requirements using supervised machine learning. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, pp. 490–495. doi:10.1109/RE.2017.82

Kurtanovic Z and Maalej W (2017b) Mining user rationale from software reviews. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017, pp. 61–70. doi:10.1109/RE.2017.86

Lange DS (2008) Text classification and machine learning support for requirements analysis using blogs. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5320 LNCS, pp. 182–195. doi:10.1007/978-3-540-89778-1_14

Li T (2018) Identifying security requirements based on linguistic analysis and machine learning. Proceedings - Asia-Pacific Software Engineering Conference, APSEC, 2017 December, pp. 388–397. doi:10.1109/APSEC.2017.45

Li C, Huang L, Ge J, Luo B and Ng V (2018) Automatically classifying user requests in crowdsourcing requirements engineering. *Journal of Systems and Software* 138, 108–123. doi:10.1016/j.jss.2017.12.028

Li J, Zhang X, Wang K, Zheng C, Tong S and Eynard B (2020) A personalized requirement identifying model for design improvement based on user profiling. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM* 34, 55–67. doi:10.1017/S0890060419000301

Lim S, Henriksson A and Zdravkovic J (2021) Data-driven requirements elicitation: a systematic literature review. *SN Computer Science*, Vol. 2. Springer Singapore. doi:10.1007/s42979-020-00416-4

Liu Y, Lu WF and Loh HT (2007) Knowledge discovery and management for product design through text mining - a case study of online information integration for designers. Proceedings of ICED 2007, the 16th International Conference on Engineering Design, DS 42(August), pp. 1–12.

Liu Y, Jin J, Ji P, Harding JA and Fung RYK (2013) Identifying helpful online reviews: a product designer's perspective. *CAD Computer Aided Design* 45, 180–194. doi:10.1016/j.cad.2012.07.008

Loper E and Bird S (2002) NLTK: the natural language toolkit. doi:10.3115/1225403.1225421

Lu M and Liang P (2017) Automatic classification of non-functional requirements from augmented app user reviews. Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, Part F1286, pp. 344–353. doi:10.1145/3084226.3084241

Lyutov A, Uygun Y and Hütt MT (2019) Managing workflow of customer requirements using machine learning. *Computers in Industry* 109, 215–225. doi:10.1016/j.compind.2019.04.010

Maalej W and Nabil H (2015) Bug report, feature request, or simply praise? On automatically classifying app reviews. Proceedings - 2015 IEEE 23rd International Requirements Engineering Conference, RE 2015, pp. 116–125. doi:10.1109/RE.2015.7320414

Maalej W and Thurimella AK (2013) Managing requirements knowledge, pp.1–20. doi: 10.1007/978-3-642-34419-0

Maalej W, Nayebi M, Johann T and Ruhe G (2015) Towards data-driven requirements engineering, pp. 1–6.

Maalej W, Kurtanović Z, Nabil H and Stanik C (2016) On the automatic classification of app reviews. *Requirements Engineering* 21, 311–331. doi:10.1007/s00766-016-0251-9

Mahmoud A (2015) An information theoretic approach for extracting and tracing non-functional requirements. Proceedings - 2015 IEEE 23rd International Requirements Engineering Conference, RE 2015, pp. 36–45. doi:10.1109/RE.2015.7320406

Maiti R and Mitropoulos F (2017) Prioritizing non-functional requirements in agile software engineering. Proceedings of the SouthEast Conference, ACMSE 2017, pp. 212–214. doi:10.1145/3077286.3077565

Manning CD and Schütze H (1999) *Foundations of Statistical Natural Language Processing*. Massachusetts Institute of Technology.

Manning CD, Raghavan P and Schütze H (2008) *Introduction to Information Retrieval Introduction*. Cambridge University Press.

Martens D and Maalej W (2019) Towards understanding and detecting fake reviews in app stores. *Empirical Software Engineering* 24, 3316–3355. doi:10.1007/s10664-019-09706-9

Massey AK, Eisenstein J, Anton AI and Swire PP (2013) Automated text mining for requirements analysis of policy documents. Proceedings - 2013 21st IEEE International Requirements Engineering Conference, RE 2013, pp. 4–13. doi:10.1109/RE.2013.6636700

Matena M, Lee K and Liu PJ (2019) Unified text-to-text transformer T5, pp. 1–53.

Meth H, Brhel M and Maedche A (2013) The state of the art in automated requirements elicitation. *Information and Software Technology* 55, 1695–1709. doi:10.1016/j.infsof.2013.03.008

Mikolov T, Chen K, Corrado G and Dean J (2013) Efficient estimation of word representations in vector space. 1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings, pp. 1–12.

Miotto R, Li L, Kidd BA and Dudley JT (2016) Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports* 6, 1–10. doi:10.1038/srep26094

Myllynen S, Suominen I, Raunio T, Karell R and Lahtinen J (2021) Developing and implementing artificial intelligence-based classifier for requirements engineering. *Journal of Nuclear Engineering and Radiation Science* 7, 1–9. doi:10.1115/1.4049722

Navarro-Almanza R, Juurez-Ramirez R and Licea G (2018) Towards supporting software engineering using deep learning: a case of software requirements classification. Proceedings - 2017 5th International Conference in Software Engineering Research and Innovation, CONISOFT 2017, 2018 January, pp. 116–120. doi:10.1109/CONISOFT.2017.00021

Noei E, Zhang F and Zou Y (2021) Too many user-reviews! What should app developers look at first? *IEEE Transactions on Software Engineering* 47, 367–378. doi:10.1109/TSE.2019.2893171

Nyamawe AS, Liu H, Niu N, Umer Q and Niu Z (2019) Automated recommendation of software refactorings based on feature requests. Proceedings of the IEEE International Conference on Requirements Engineering, 2019 September, pp. 187–198. doi:10.1109/RE.2019.00029

Ormandjieva O, Hussain I and Kosseim L (2007) Toward a text classification system for the quality assessment of software requirements written in natural language. SOQUA'07: Fourth International Workshop on Software Quality Assurance - In Conjunction with the 6th ESEC/FSE Joint Meeting, pp. 39–45. doi:10.1145/1295074.1295082

Ott D (2013) Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7830 LNCS, pp. 50–64. doi:10.1007/978-3-642-37422-7_4

Pan S and Ding T (2019) Social media-based user embedding: a literature review. IJCAI International Joint Conference on Artificial Intelligence, 2019 August, pp. 6318–6324. doi:10.24963/ijcai.2019/881

Panichella S, Di Sorbo A, Guzman E, Visaggio CA, Canfora G and Gall HC (2015) How can I improve my app? Classifying user reviews for software maintenance and evolution. Proceedings - 2015 IEEE 31st International Conference on Software Maintenance and Evolution, ICSME 2015, September, pp. 281–290. doi:10.1109/ICSM.2015.7332474

Parra E, Dimou C, Llorens J, Moreno V and Fraga A (2015) A methodology for the classification of quality of requirements using machine learning techniques. *Information and Software Technology* 67, 180–195. doi:10.1016/j.infsof.2015.07.006

Pedregosa F, Varoquaux G, Alexandre G and Thirion B (2011) Scikit-learn: machine learning in python. *Journal of Machine Learning Research* 12, 2825–2830. doi:10.1289/EHP4713

Pennington J, Socher R and Manning CD (2014) GloVe: global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing.

Perez-Verdejo JM, Sanchez-Garcia AJ and Ocharan-Hernandez JO (2020) A systematic literature review on machine learning for automated requirements classification. Proceedings - 2020 8th Edition of the International Conference in Software Engineering Research and Innovation, CONISOFT 2020, pp. 21–28. doi:10.1109/CONISOFT50191.2020.00014

Perini A (2018) Data-driven requirements engineering. The SUPERSEDE Way. Information Management and Big Data, 1(Annual International Symposium on Information Management and Big Data), pp. 13–18. doi:10.1007/978-3-030-11680-4

Petcuşin F, Stănescu L and Bădică C (2020) An experiment on automated requirements mapping using deep learning methods. *Studies in Computational Intelligence* 868, 86–95. doi:10.1007/978-3-030-32258-8_10

Polpinij J and Namee K (2021) Automatically retrieving of software specification requirements related to each actor. *Lecture Notes in Networks and Systems*, Vol. 251. Springer International Publishing. doi:10.1007/978-3-030-79757-7_12

Prasetyo PK, Lo D, Achananuparp P, Tian Y and Lim EP (2012) Automatic classification of software related microblogs. IEEE International Conference on Software Maintenance, ICSM, pp. 596–599. doi:10.1109/ICSM.2012.6405330

Provost F and Fawcett T (2013) Data science and its relationship to big data and data-driven decision making. *Big Data* 1, 51–59. doi:10.1089/big.2013.1508

Qi J, Zhang Z, Jeon S and Zhou Y (2016) Mining customer requirements from online reviews: a product improvement perspective. *Information and Management* 53, 951–963. doi:10.1016/j.im.2016.06.002

Rahimi N, Eassa F and Elrefaei L (2020) An ensemble machine learning technique for functional requirement classification. *Symmetry* 12, 1601.

Rahman MA, Haque MA, Tawhid MNA and Siddik MS (2019) Classifying non-functional requirements using RNN variants for quality software development. MaLTeSQuE 2019 - Proceedings of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation, Co-Located with ESEC/FSE 2019, pp. 25–30. doi:10.1145/3340482.3342745

Rashwan A, Ormandjieva O and Witte R (2013) Ontology-based classification of non-functional requirements in software specifications: a new corpus and SVM-based classifier. Proceedings - International Computer Software and Applications Conference, Vol. ii, pp. 381–386. doi:10.1109/COMPSAC.2013.64

Riaz M, King J, Slankas J and Williams L (2014) Hidden in plain sight: automatically identifying security requirements from natural language artifacts. Proceedings - 2014 IEEE 22nd International Requirements Engineering Conference, RE 2014, 183–192. doi:10.1109/RE.2014.6912260

Rodeghero P, Jiang S, Armaly A and McMillan C (2017) Detecting user story information in developer-client conversations to generate extractive summaries. Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE 2017, pp. 49–59. doi:10.1109/ICSE.2017.13

Sabir M, Banissi E and Child M (2021) A deep learning-based framework for the classification of non-functional requirements. *Advances in Intelligent Systems and Computing: Vol. 1366 AISC*. Springer International Publishing. doi:10.1007/978-3-030-72651-5_56

Sampada GC, Sake TI and Chhabra M (2020) A review on advanced techniques of requirement elicitation and specification in software development stages. PDGC 2020 - 2020 6th International Conference on Parallel, Distributed and Grid Computing, pp. 215–220. doi:10.1109/PDGC50313.2020.9315741

Shabestari SS, Herzog M and Bender B (2019) A survey on the applications of machine learning in the early phases of product development. Proceedings of the International Conference on Engineering Design, ICED, 2019 August, pp. 2437–2446. doi:10.1017/dsi.2019.250

Singh A and Tucker CS (2017) A machine learning approach to product review disambiguation based on function, form and behavior classification. *Decision Support Systems* 97, 81–91. doi:10.1016/j.dss.2017.03.007

Stanik C, Haering M and Maalej W (2019) Classifying multilingual user feedback using traditional machine learning and deep learning. Proceedings - 2019 IEEE 27th International Requirements Engineering Conference Workshops, REW 2019, pp. 220–226. doi:10.1109/REW.2019.00046

Stone T and Choi SK (2013) Extracting consumer preference from user-generated content sources using classification. Proceedings of the ASME Design Engineering Technical Conference, 3 A, pp. 1–9. doi:10.1115/DETC2013-13228

Suryadi D and Kim HM (2019) A data-driven approach to product usage context identification from online customer reviews. *Journal of Mechanical Design, Transactions of the ASME* 141, 1–13. doi:10.1115/1.4044523

Sutskever I, Vinyals O and Le QV (2014) Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems* **4**, 3104–3112.

Taj S, Arain Q, Memon I and Zubedi A (2019) To apply data mining for classification of crowd sourced software requirements. PervasiveHealth: Pervasive Computing Technologies for Healthcare, pp. 42–46. doi:10.1145/3328833.3328837

Tamai T and Anzai T (2018) Quality requirements analysis with machine learning. ENASE 2018 - Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering, 2018 March, pp. 241–248. doi:10.5220/0006694502410248

Tang M, Jin J, Liu Y, Li C and Zhang W (2019) Integrating topic, sentiment, and syntax for modeling online reviews: a topic model approach. *Journal of Computing and Information Science in Engineering* **19**, 1–12. doi:10.1115/1.4041475

Timoshenko A and Hauser JR (2019) Identifying customer needs from user-generated content. *Marketing Science* **38**, 1–20. doi:10.1287/mksc.2018.1123

Tóth L and Vidács L (2018) Study of various classifiers for identification and classification of non-functional requirements. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10964 LNCS, pp. 492–503. doi:10.1007/978-3-319-95174-4_39

US DoD (2018) *Digital Engineering Strategy.* Office of the Deputy Assistant Secretary of Dense for Systems Engineering.

Vinyals O, Toshev A, Bengio S and Erhan D (2015) Show and tell: a neural image caption generator. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12 June, pp. 3156–3164. doi:10.1109/CVPR.2015.7298935

Wang C, Zhang F, Liang P, Daneva M and Van Sinderen M (2018*a*) Can app changelogs improve requirements classification from app reviews?: an exploratory study. *International Symposium on Empirical Software Engineering and Measurement.* doi:10.1145/3239235.3267428

Wang W, Feng Y and Dai W (2018*b*) Topic analysis of online reviews for two competitive products using latent Dirichlet allocation. *Electronic Commerce Research and Applications* **29**, 142–156. doi:10.1016/j.elerap.2018.04.003

Wang Z, Chen CH, Zheng P, Li X and Khoo LP (2019) A novel data-driven graph-based requirement elicitation framework in the smart product-service system context. *Advanced Engineering Informatics* **42**, 100983. doi:10.1016/j.aei.2019.100983

Wiegers K and Beatty J (2013) *Software Requirements,* 3rd Edn. Microsoft Press.

Winkler J and Vogelsang A (2017) Automatic classification of requirements based on convolutional neural networks. Proceedings - 2016 IEEE 24th International Requirements Engineering Conference Workshops, REW 2016, pp. 39–45. doi:10.1109/REW.2016.16

Wong LR, Mauricio D and Rodriguez GD (2017) A systematic literature review about software requirements elicitation. *Journal of Engineering Science and Technology* **12**, 296–317.

Xie H, Yang J, Chang CK and Liu L (2017) A statistical analysis approach to predict user's changing requirements for software service evolution. *Journal of Systems and Software* **132**, 147–164. doi:10.1016/j.jss.2017.06.071

Xu F, Uszkoreit H, Du Y, Fan W and Zhao D (2019) Explainable AI: a brief survey on history, research areas, approaches and challenges. CCF International Conference on Natural Language Processing and Chinese Computing. doi:10.1007/978-3-030-32236-6

Yan X, Guo J, Lan Y and Cheng X (2013) A biterm topic model for short texts. Proceedings of the 22nd International Conference on World Wide Web, pp. 1445–1455.

Yang B, Liu Y, Liang Y and Tang M (2019) Exploiting user experience from online customer reviews for product design. *International Journal of Information Management* **46**, 173–186. doi:10.1016/j.ijinfomgt.2018.12.006

Yao L, Torabi A, Cho K, Ballas N, Pal C, Larochelle H and Courville A (2015) Describing videos by exploiting temporal structure. Proceedings of the IEEE International Conference on Computer Vision, pp. 4507–4515.

Young RR (2004) *The Requirements Engineering Handbook.* London: Artech House.

Zhou F, Ayoub J, Xu Q and Yang XJ (2020) A machine learning approach to customer needs analysis for product ecosystems. *Journal of Mechanical Design, Transactions of the ASME* **142**, 1–14. doi:10.1115/1.4044435

Zimmerman P (2017) DoD digital engineering strategy. 20th Annual NDIA Systems Engineering Conference, Springfield, VA, USA, October 23–26, 2017.

Cheligeer Cheligeer completed his Ph.D. program at Concordia University's Institute for Information Systems Engineering (CIISE) in 2022. He received his Master of Information Technology and Master of Business Information System from Monash University, Australia in 2015. His research interests include artificial intelligence, natural language processing, system design, software engineering, and health data science.

Jingwei Huang is a Ph.D. in Information Engineering (University of Toronto, 2008) and an Associate Professor of Systems Engineering at Old Dominion University. He has extensive research experience applying artificial intelligence in systems engineering and digital trust. His previous work includes using a logic-based AI approach for Knowledge Provenance ontologies, formal semantics of trust, trust calculus, trust mechanisms, trust models in PKIs, security policies integrating role-based and attribute-based access control models, and their applications in industry. Dr. Huang's current research focuses on Digital Systems Engineering to develop trustworthy AI/ML & Big Data solutions for Digital Engineering Transformation.

Guosong Wu is a postdoc fellow at the Centre for Health Informatics, Cumming School of Medicine, University of Calgary and Institute of Health Economics, University of Alberta. He received his Ph.D. in Health Services Research at the Department of Critical Care Medicine, University of Calgary, focused on the healthcare system performance evaluation by developing indicators to measure the quality and safety of care. His research interests focus on the improvement of healthcare quality through machine learning driven analytics of structured and unstructured Electronic Medical Records and administrative databases.

Nadia Bhuiyan obtained her bachelor's degree in Industrial Engineering from Concordia University and her M.Sc. and Ph.D. in Mechanical Engineering from McGill University. Her research focuses on product development processes, dealing with the design, development, production, and distribution of goods and services. Dr. Bhuiyan served as Associate Director and Director of Education of the Concordia Institute of Aerospace Design and Innovation, and Director of the Master of Aerospace Engineering program. She currently serves as Vice-Provost of Partnerships and Experiential Learning.

Yuan Xu is an Assistant Professor in Departments of Oncology, Community Health Science, and Surgery at University of Calgary. He obtained his M.D. and M.Sc. degrees and general surgery residency at Capital Medical University in Beijing, China. Subsequently, he received trainings on epidemiology and health services research as Ph.D. and Postdoctoral fellow at University of Calgary. Dr. Xu's main interest lies in developing methods to leverage multidimensional health data including electronic health data and administrative data to boost the outcome of health service studies, and to generate real-world evidence to support medical decision making at the health-system and point-care levels in cancer care.

Yong Zeng is a Professor in Information Systems Engineering at Concordia University. He is the President of the Society for Design and Process Science. He was NSERC Chair in aerospace design engineering (2015–2019) and Canada Research Chair in design science (2004–2014). Zeng researches into creative design by developing and employing mathematical and neurocognitive approaches. He has proposed Environment-Based Design (EBD) addressing the recursive nature of design and the role of mental stress in designer creativity. He applies the EBD to aerospace industry, medical device design, human resource management, municipality, teaching and learning design, and health.

Appendix

See

Table A1. The included works (#: study id)

#	Title	Study reference	#	Title	Study reference
S01	Study of various classifiers for identification and classification of non-functional requirements	(Tóth and Vidács, 2018)	S02	An information theoretic approach for extracting and tracing non-functional requirements	(Mahmoud, 2015)
S03	Classifying non-functional requirements using RNN variants for quality software development	(Rahman et al., 2019)	S04	Mining user rationale from software reviews	(Kurtanovic and Maalej, 2017b)
S05	Automated text mining for requirements analysis of policy documents	(Massey et al., 2013)	S06	Towards supporting software engineering using deep learning: A case of software requirements classification	(Navarro-Almanza et al., 2018)
S07	Automatic classification of non-functional requirements from augmented app user reviews	(Li et al., 2018)	S08	Can app changelogs improve requirements classification from app reviews? An exploratory study	(Wang et al., 2018)
S09	Automatic classification of non-functional requirements from augmented app user reviews	(Lu and Liang, 2017)	S10	Integrating topic, sentiment, and syntax for modeling online reviews: A topic model approach	(Tang et al., 2019)
S11	Hidden in plain sight: Automatically identifying security requirements from natural language artifacts	(Riaz et al., 2014b)	S12	A comprehensive characterization of NLP techniques for identifying equivalent requirements	(Falessi et al., 2010)
S13	Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements	(Ott, 2013)	S14	Automated identification of security requirements: A ML approach	(Kobilica et al., 2020)
S15	Automatic requirements elicitation from social media (ARESM)	(Kengphanphanit and Muenchaisri, 2020)	S16	Requirements classification with interpretable ML and dependency parsing	(Dalpiaz et al., 2019)
S17	Aspect and sentiment unification model for online review analysis	(Jo and Oh, 2011)	S18	Using recurrent neural networks for classification of natural language-based non-functional requirements	(Gnanasekaran et al., 2021)
S19	Exploiting user experience from online customer reviews for product design	(Yang et al., 2019)	S20	Software requirements classification using ML algorithms	(Canedo and Mendes, 2020)
S21	Automated keyword filtering in latent Dirichlet allocation for identifying product attributes from online reviews	(Joung and Kim, 2021)	S22	App review analysis via active learning: Reducing supervision effort without compromising classification accuracy	(Dhinakaran et al., 2018)
S23	Towards understanding and detecting fake reviews in app stores	(Martens and Maalej, 2019)	S24	Identification of non-functional requirements in textual specifications: A semi-supervised learning approach	(Casamayor et al., 2010)
S25	Eliciting attribute-level user needs from online reviews with deep language models and information extraction	(Han and Moghaddam, 2021)	S26	RE Data challenge: Requirements Identification with Word2Vec and TensorFlow	(Dekhtyar and Fong, 2017)
S27	A methodology for the classification of quality of requirements using ML techniques	(Parra et al., 2015)	S28	Automatically classifying functional and non-functional requirements using supervised ML	(Kurtanovic and Maalej, 2017a)
S29	Identifying helpful online reviews with word embedding features	(Chen et al., 2016)	S30	A ML approach to customer needs analysis for product ecosystems	(Zhou et al., 2020)
S31	AR-miner: Mining informative reviews for developers from mobile app marketplace	(Chen et al., 2014)	S32	Classifying multilingual user feedback using traditional ML and deep learning	(Stanik et al., 2019)
S33	Extracting customer perceptions of product sustainability from online reviews	(El Dehaibi et al., 2019)	S34	An experiment on automated requirements mapping using deep learning methods	(Petcuşin et al., 2020)
S35	Mining customer requirements from online reviews: A product improvement perspective	(Qi et al., 2016)	S36	Too many user-reviews! What should App developers look at first?	(Noei et al., 2021)
S37	Using classification techniques	(Ko et al., 2007)	S38	Classification of the	(Jeon et al.,

	for informal requirements in the requirements analysis-supporting system			requirement sentences of the US DOT standard specification using deep learning algorithms	2021)
S39	What works better? A study of classifying requirements	(Abad *et al.*, 2017)	S40	Automatic classification of requirements based on convolutional neural networks	(Winkler and Vogelsang, 2017)
S41	Systematic service product requirement analysis with online customer review data	(Jones and Kim, 2015)	S42	An approach to clustering and sequencing of textual requirements	(Barbosa *et al.*, 2015)
S43	Identifying helpful online reviews: A product designer's perspective	(Liu *et al.*, 2013)	S44	Which feature is unusable? Detecting usability and user experience issues from user reviews	(Bakiu and Guzman, 2017)
S45	To apply data mining for classification of crowd sourced software requirements	(Taj *et al.*, 2019)	S46	Automated classification of non-functional requirements	(Cleland-Huang *et al.*, 2007)
S47	A deep learning-based framework for the classification of non-functional requirements	(Sabir *et al.*, 2021)	S48	Automatic multi-class non-functional software requirements classification using neural networks	(Baker *et al.*, 2019)
S49	Towards a software requirements change classification using support vector machine	(Khelifa *et al.*, 2018)	S50	Automated classification of security requirements	(Jindal *et al.*, 2016)
S51	Developing and implementing artificial intelligence-based classifier for requirements engineering	(Myllynen *et al.*, 2021)	S52	Topic modeling on user stories using word mover's distance	(Gulle *et al.*, 2020)
S53	Ontology-based classification of non-functional requirements in software specifications: A new corpus and SVM-based classifier	(Rashwan *et al.*, 2013)	S54	Extracting consumer preference from user-generated content sources using classification	(Stone and Choi, 2013)
S55	Quality requirements analysis with ML	(Tamai and Anzai, 2018)	S56	Identifying security requirements based on linguistic analysis and ML	(Li, 2018)
S57	Managing workflow of customer requirements using ML	(Lyutov *et al.*, 2019)	S58	How do users like this feature? A fine grained sentiment analysis of App reviews	(Guzman and Maalej, 2014)
S59	A data-driven approach to product usage context identification from online customer reviews	(Suryadi and Kim, 2019)	S60	Using linguistic knowledge to classify non-functional requirements in SRS documents	(Hussain *et al.*, 2008)
S61	Automated recommendation of software refactorings based on feature requests	(Nyamawe *et al.*, 2019)	S62	Automated recommendation of software refactorings based on feature requests	(Casamayor *et al.*, 2009)
S63	Detecting non-atomic requirements in software requirements specifications using classification methods	(Halim and Siahaan, 2019)	S64	Topic analysis of online reviews for two competitive products using latent Dirichlet allocation	(Wang *et al.*, 2018)
S65	How can I improve my app? Classifying user reviews for software maintenance and evolution	(Panichella *et al.*, 2015)	S66	Analysis of user comments: An approach for software requirements evolution	(Carreno and Winbladh, 2013)
S67	Non-functional requirements classification with feature extraction and ML: An empirical study	(Haque *et al.*, 2019)	S68	Why people hate your App: Making sense of user feedback in a mobile app store	(Fu *et al.*, 2013)
S69	A little bird told me: Mining tweets for requirements and software evolution	(Guzman *et al.*, 2017)	S70	Toward a text classification system for the quality assessment of software requirements written in natural language	(Ormandjieva *et al.*, 2007)
S71	Detecting user story information in developer-client conversations to generate	(Rodeghero *et al.*, 2017)	S72	Identifying customer needs from user-generated content	(Timoshenko and Hauser, 2019)

	extractive summaries				
S73	A ML approach to product review disambiguation based on function, form, and behavior classification	(Singh and Tucker, 2017)	S74	Knowledge discovery and management for product design through text mining: A case study of online information integration for designers	(Liu *et al.*, 2007)
S75	Text classification and ML support for requirements analysis using blogs	(Lange, 2008)	S76	A personalized requirement identifying model for design improvement based on user profiling	(Li *et al.*, 2020)
S77	Automatically retrieving of software specification requirements related to each actor	(Polpinij and Namee, 2021)	S78	Automatically classifying non-functional requirements with feature extraction and supervised ML techniques: A research preview	(EzzatiKarami and Madhavji, 2021)
S79	Mining non-functional requirements from App store reviews	(Jha and Mahmoud, 2019)	S80	Data-driven elicitation and optimization of dependencies between requirements	(Deshpande *et al.*, 2019)
S81	Clustering mobile Apps based on mined textual features	(Al-Subaihin *et al.*, 2016)	S82	An ensemble ML technique for functional requirement classification	(Rahimi *et al.*, 2020)
S83	Annotation of software requirements specification (SRS), extractions of nonfunctional requirements, and measurement of their tradeoff	(Asif *et al.*, 2019)	S84	A ML-based approach for demarcating requirements in textual specifications	(Abualhaija *et al.*, 2019)
S85	On the automatic classification of app reviews	(Walid Maalej *et al.*, 2016)	S86	Automatic classification of software related microblogs	(Prasetyo *et al.*, 2012)

Table A2. The extracted tasks for the included works

Category	Task type	Identified works	Count
Preparation	Product knowledge acquisition	S42, S52, S59, S71, S74, S81	6
	UGC helpfulness prediction	S29, S35, S43	3
	UGC categorization	S04, S08, S22, S32, S44, S65, S69, S73, S75, S85, S86	11
	User preference extraction	S10, S17, S19, S21, S25, S30, S31, S33, S36, S41, S54, S58, S64, S66, S68, S72	16
Collection	Requirement identification	S05, S07, S15, S38, S40, S84	6
	Requirement classification	S13, S16, S20, S26, S28, S34, S37, S39, S45, S51, S55, S60, S62, S63, S76, S77, S83	17
	NFR/FR classification	S01, S02, S03, S06, S09, S18, S24, S46, S47, S48, S53, S67, S78, S79, S82	15
	Security requirement identification	S11, S14, S50, S56	4
Inspection	Equivalent detection, fake review detection, dependency extraction, quality assessment	S12, S23, S27, S70, S80	5
Negotiation	Requirement change support, requirement distribution, refactoring decision support	S49, S57, S61	3

Table A3. The data source categorization

Category	Data type	Identified works
Available documents	Requirement specifications	S7, S11, S34, S37, S38, S42, S51, S55, S63, S70, S77, S82
	User stories	S52, S71
	Research articles	S74
	Policy document	S5
	Real industrial data	S2, S12, S13, S40, S57, S84
Existing Requirement dataset	DePaul NFR dataset	S1, S3, S6, S16, S18, S20, S24, S26, S28, S39, S46, S47, S48, S49, S50, S53, S60, S62, S67, S78, S83
	SecReq dataset	S14, S26, S56
	PURE corpus	S78, S80
	INCOSE corpus	S27
User-Generated Content	E-commerce UGC	S4, S10, S17, S19, S25, S29, S30, S33, S35, S43, S44, S59, S64, S72, S73, S76
	Mobile app review UGC	S8, S9, S21, S22, S23, S31, S36, S58, S65, S66, S68, S79, S85
	Micro-blog, social media	S15, S32, S54, S69, S86,
	Others	S41, S45, S61, S75

Table A4. The applied textual features

Category	Features	Identified works
Bag-of-words	N-gram lexical features with count weighting	S4, S15, S16, S17, S21, S22, S25, S30, S33, S36, S41, S49, S52, S53, S57, S58, S64, S66, S67, S68, S73, S74, S77, S78, S80, S85, S86
	N-gram with TF-IDF weighting	S1, S5, S7, S9, S12, S20, S24, S29, S32, S34, S37, S41, S44, S46, S50, S61, S62, S67, S69, S76, S78, S79, S81, S82, S83
Rule-based features	Linguistic features	S4, S16, S27, S28, S32, S39, S43, S56, S60, S63, S70, S84
	Frequency-based features	S4, S16, S27, S28, S32, S35, S43, S70, S71
	Sentiment features	S4, S16, S32, S43
	Temporal features	S32, S39, S43
	Domain keyword features	S27, S32, S56, S60, S84
	Requirement quality features	S27, S35, S43, S70
	Meta-data features	S4, S16, S23, S35, S85
Embedding features	Pretrained word vectors	S3, S18, S26, S30, S34, S38, S52
	Embedding trained from scratch	S6, S29, S38, S40, S47, S48
	Fine-tuned embeddings	S25, S47

Table A5. The applied learning algorithms

Category	Algorithms	Identified works
Supervised learning	Naïve Bayes	S1, S4, S7, S8, S9, S11, S13, S15, S19, S20, S22, S23, S24, S31, S37, S41, S45, S56, S57, S61, S62, S63, S65, S67, S69, S73, S75, S77, S78, S79, S82, S83, S85
	Support vector machine	S1, S4, S7, S11, S13, S14, S16, S20, S22, S23, S28, S41, S44, S49, S53, S56, S57, S61, S65, S67, S71, S73, S74, S78, S79, S80, S82, S84, S86
	Decision tree-based algorithms	S1, S8, S9, S14, S23, S27, S39, S45, S50, S56, S57, S60, S63, S65, S67, S70, S73, S78, S82, S83, S84, S85
	Logistic regression	S1, S4, S12, S20, S22, S33, S56, S61, S65, S71, S78, S82, S83, S84
	Convolutional neural networks	S3, S6, S14, S25, S26, S32, S38, S40, S47, S48, S55, S72
	Feedforward neural networks	S1, S23, S34, S38, S43, S47, S51, S57, S59
	K-nearest neighbor	S1, S8, S11, S14, S20, S61, S67, S73
	Random forest	S1, S23, S73, S78, S84
	Recurrent neural networks	S3, S14, S18, S38, S47
	Regression	S29, S35, S43
Unsupervised learning	Topic modeling methods	S5, S10, S17, S21, S30, S31, S36, S52, S58, S64, S66, S68, S69, S76
	Clustering methods	S2, S42, S59, S81