Identifying Implicit Requirements in SRS Big Data

Conference Paper · December 2019

DOI: 10.1109/BigData47090.2019.9006086

3 authors:

Onyeka Emebo
Covenant University Ota Ogun State, Nigeria
18 PUBLICATIONS **95** CITATIONS
SEE PROFILE

Vaibhav Anu
Montclair State University
28 PUBLICATIONS **117** CITATIONS
SEE PROFILE

Aparna S. Varde
Montclair State University
125 PUBLICATIONS **662** CITATIONS
SEE PROFILE

Identifying Implicit Requirements in SRS Big Data

Emebo Onyeka*, Vaibhav Anu†, Aparna S. Varde‡
Covenant University (Ota, Nigeria)*, Montclair State University (Montclair, USA)†‡
onye.emebo@covenantuniversity.edu.ng*, anuv@montclair.edu†, vardea@montclair.edu‡

***Abstract*—Over the past few years, we have worked on pioneering an approach that employs Commonsense Knowledge (CSK) to automate the identification of Implicit Requirements (IMRs) from text in large Software Requirements Specifications (SRS) documents. This paper builds on our IMR-identification approach by adding CNN-based deep learning to detect IMRs from complex SRS big data such as images and tables.**

***Index Terms*—Commonsense Knowledge, Domain Ontology, IMRs, Requirements Engineering, Text Mining**

I. Problem Definition

Requirements engineering (RE) is a systematic process with many activities. Elicitation is an RE activity wherein requirements are collected from stakeholders. Requirements can be classified as explicit and implicit. Explicit requirements are clearly stated by the business users during elicitation. Implicit requirements (IMRs), on the other hand, are assumed/hidden yet crucial requirements that a system is expected to fulfil (although not explicitly stated by the users). Whilst not directly captured during elicitation, IMRs have a significant impact in the success or failure of software development [1].

Organizations usually rely on human analysts to manually read through the SRS documents to identify any IMRs related to the explicitly stated requirements recorded in the SRS documents. However, depending on the size and complexity of the software system being developed, there are huge amounts of requirements data in SRS documents. Manual scanning of these by human analysts to identify IMRs is tedious, infeasible, and non-scalable, with growing big data. Thus, there is an imperative need for automating the IMR-identification process.

II. Proposed Solution

Our work addresses the need for automating the IMR-identification process to ensure successful software development. In our previous work [2], we proposed a framework called **COTIR**: **C**ommonsense knowledge **O**ntology and **T**ext mining for **I**mplicit **R**equirements, addressing plain text in SRS. In the current paper, we propose an enhanced COTIR approach with Convolutional Neural Network (CNN) based autoencoders in lieu of heuristic classification. Figure 1 summarizes our solution.

A. Functioning of Fundamental COTIR Framework

The left side of Figure 1 shows the fundamental (i.e. existing) COTIR framework while its right side shows the proposed enhancements to COTIR. In *fundamental COTIR*, source SRS are first converted to requirements in text format (without images, tables etc.); relevant CSK Knowledge Bases (CSKBs) developed from a source called *WebChild [3]* are considered next. SRS documents, CSKBs and domain ontology (selected by SRS authors) are transferred to the *Feature Extraction* module by which possible IMR sources are outlined. *Heuristic Classification* then identifies potential IMRs. For more details on *fundamental COTIR*, please see [2].

B. Functioning of Enhanced COTIR Framework

In *enhanced COTIR*, a convolutional autoencoder is used instead of a heuristic classifier (see Fig.1 right). This enables COTIR to: (1) process complex data in SRS such as images and find subtle IMRs, (2) learn more meaningful representations of textual data compared to heuristic classifiers. This leverages big data *Vs*, entailing *variety* in addition to *volume*.

The functioning of our proposed enhanced COTIR framework can be described as the following sequence of steps:

Step 1: Requirements documents (SRS) supply the requirements data from which IMRs need to be identified. RE data preprocessing removes the noise in the RE data. This step is performed by the NLP Processor component (see Fig. 1).

Step 2: The requirements author then selects the relevant knowledge base from CSKBs and the relevant domain ontology from the Ontology Library. The previously preprocessed RE data along with the selected KB and domain ontology are transferred to the CNN-based autoencoder component (Convolutional Encoder-Decoder in Fig. 1).

Step 3: The autoencoder's input construction transforms RE data into vectors for deep learning models. For IMR-identification in SRS, frequency is calculated for those words that make a requirement statement implicit, and the word frequency values are transformed into vectors. These vectors are regarded as a training set for the autoencoder. Model training with deep learning is then required on the designed model with the given training set. A deep learning model can contain thousands of parameters depicting weights of connections among neural units. Thus, training the model entails tuning parameters based on the training set. In an autoencoder, parameters are trained by minimizing differences between the input and output layers in an unsupervised manner.

Step 4: Next, the trained model is applied to solve new IMR-problems. This trained model is capable of encoding the word frequency vector of a new IMR feature into the hidden states. We can trace and calculate changes of values in each vector dimension along with the encoding process, and then deduce weights of words in each dimension. These word weights are

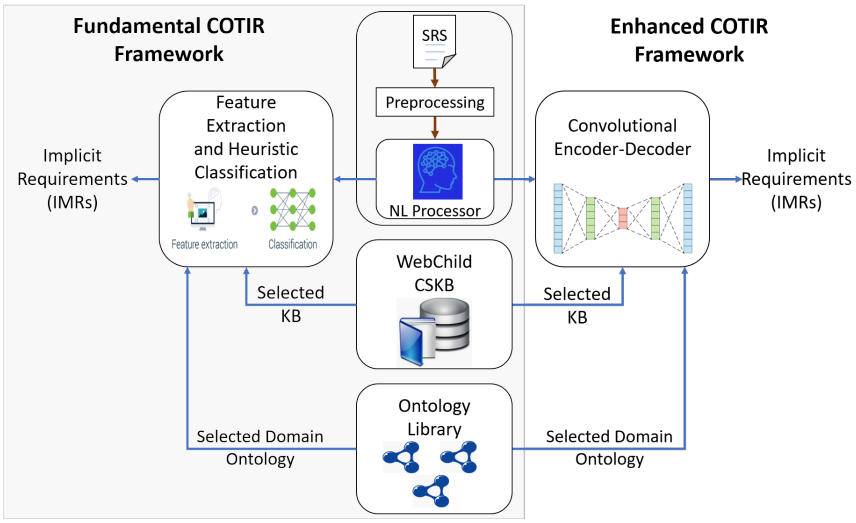


Fig. 1. COTIR Framework: Fundamental and Enhanced

expected to help in assigning weights to the sentences and select informative ones.

III. Preliminary Evaluation

In this section, we provide the results from evaluating the fundamental COTIR framework. For our evaluation, we compared the IMR-identification performance of fundamental COTIR framework with that of 8 software engineering experts (SE researchers and IT professionals). The following subsections describe the requirements artifacts used in the study, study design, and study results.

A. Study Artifacts

The following 3 artifacts were used in the study: i) SRS-1: Software requirements specifications for a Course Management System, ii) SRS-2: Software requirements specifications for an Embedded Monitoring Project, and iii) SRS-3: Software requirements specifications for a Tactical Control System.

B. Study Design

Study participants were asked to assess implicitness in 3 SRS documents and to use the COTIR tool. When identifying IMRs manually, the 8 participants were supplied with a report form shown in Fig. 2. They were given the following instructions: 1) for each specified requirement, mark the requirement based on its implicit nature (noting that a requirement may contain more than one form of implicitness as in Fig. 2); and 2) For each requirement, specify the degree of criticality of each implicitness on a scale of 1 to 5 (1 being least critical to 5 being the most). In this paper, we limit our discussion to implicitness-identification aspects of the study.

C. Study Results

The 8 study participants first carefully read each of the 3 SRS documents supplied to them and identified requirements with implicit patterns. As a result of this manual effort, all participants collectively were able to identify 8 potential IMRs

#	Requirement	Type of Implicitness	Criticality				
1	The C&C shall provide the users with real-time data regarding the measured values, as collected from the various sensors that are part of the network	Ambiguity	1	2	3	4	5
		Incomplete Knowledge	1	2	3	4	5
		Vagueness	1	2	3	4	5
		Miscellaneous	1	2	3	4	5
2	The C&C shall support the configuration of ranges of sensor readings (maximum and minimum allowed values)	Ambiguity	1	2	3	4	5
		Incomplete Knowledge	1	2	3	4	5
		Vagueness	1	2	3	4	5
		Miscellaneous	1	2	3	4	5
3	The C&C shall notify users if there are manually modified values, whenever it presents sensor data to them.	Ambiguity	1	2	3	4	5
		Incomplete Knowledge	1	2	3	4	5
		Vagueness	1	2	3	4	5
		Miscellaneous	1	2	3	4	5

Fig. 2. COTIR Evaluation: Sample Form

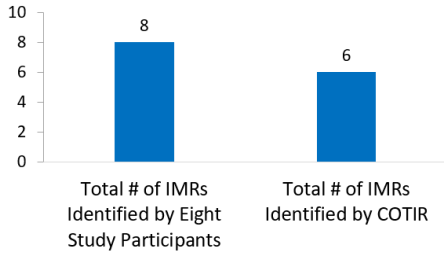
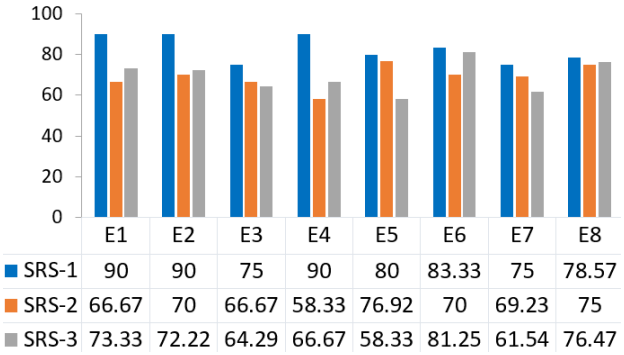


Fig. 3. IMRs Identfied by COTIR vs Human Analysts

in the 3 SRS documents. Next, the COTIR tool was used to identify the following types of implicit patterns in the 3 SRS documents: i) Ambiguity, ii) Incomplete Knowledge, iii) Vagueness, and iv) Miscellaneous. The experts' evaluation served as the ground truth.

A major result of this study was that the COTIR approach was able to identify 6 out of 8 known instances of implicit patterns in the supplied requirements (see Fig. 3). This is a significant result as it provides evidence that COTIR can relieve human analysts from the tedious manual task of reading huge SRS documents to find IMRs.

We also computed the recall, precision and F-scores for the COTIR tool relative to the 8 experts' evaluations (that



	E1	E2	E3	E4	E5	E6	E7	E8
SRS-1	90	90	75	90	80	83.33	75	78.57
SRS-2	66.67	70	66.67	58.33	76.92	70	69.23	75
SRS-3	73.33	72.22	64.29	66.67	58.33	81.25	61.54	76.47

Fig. 4. COTIR Evaluation: Recall Metrics

constitute ground truth). For an IMR identification tool, recall is more significant than precision. This is because it is very important to try and find *all* requirements that are implicit (ideally 100% recall). If any of them are left out, it is more problematic than the opposite situation of stating that something is an IMR when it is not. In other words, false negatives in this task are more crucial than false positives, emphasizing greater significance of recall. Hence, we focus on recall here. The recall values computed for the COTIR tool relative to 8 experts' evaluations are shown in Fig. 4. In the ideal case, recall should be 100%, as it would completely relieve human analysts from manually analyzing SRS documents to identify IMRs. During our study, the average recall was found to be: 82.74% for SRS-1, 69.1% for SRS-2, and 69.26% for SRS-3. The combined average recall for the 3 SRS documents was computed to be 73.7%.

Hence, COTIR yielded a recall value of 73.7% on the whole (see Fig. 4). Accordingly, it was found that the performance of our IMR-identification approach COTIR was consistent with best practices in software engineering and that the tool was considered usable as determined by software engineers. They indicated this through their verbal comments as well.

IV. Related Work

Different researchers have addressed issues related to IMRs. The work in [7] presents a model to compute similarities between SRS to promote their analogical reuse. Hence, requirement reuse is based on the detection of analogies in specifications. Though identification of analogies in requirements is essential, this study does not discuss management of IMRs. In [6], there is prototype tool developed called SR-Elicitor which is an Eclipse plugin and can be used by software engineers to transform natural language software requirements to SRS based on a Semantic of Business Vocabulary and Rules (SBVR), a recent standard. In [5], authors analyze approaches that aim to address ambiguities in natural language SRS, focusing on state-of-the-art tools for ambiguity resolution such as NAI and ARUgen (in addition to SR-Elicitor). While these are signficiant advances in IMR research, comparative studies [4] indicate that COTIR outperforms these tools for IMR identification in some respects.

The deployment of CSK is useful in AI systems [8]. Notable CSK sources include Cyc, WordNet and WebChild [3], [9], [10]. Potential use of CSK to enhance object detection appears in [11] and its application to autonomous vehicles is explained in [12], while [13] and [14] outline the role of CSK in mining ordinances and their public reactions using human judgment. In our COTIR work, CSK usage stands out given the fact that we consider it in conjunction with ontology and text mining. As far as we know, ours is among the first works to harness CSK in an IMR context. This is a striking aspect of COTIR research, embodied in fundamental and enhanced COTIR.

V. Conclusions

IMRs in SRS are crucial to the success of software development. Researchers have addressed IMRs considering various facets. Our work stands out as it *introduces commonsense knowledge for IMRs*. Our *COTIR* approach addresses big data on SRS via CSK to identify IMRs. Comparative studies with state-of-the-art show superior performance of COTIR [4]. Motivated by successful evaluation of *fundamental COTIR*, we present *enhanced COTIR* herewith, adding deep learning with CNNs to identify IMRs from tables and images in large SRS documents. To the best of our knowledge, this paper is the first to *propose CNNs in IMR-identification*. Detailed studies on enhancements constitute part of our ongoing work.

References

[1] V. Gervasi, R. Gacitua, M. Rouncefield, P. Sawyer, et al. Unpacking Tacit Knowledge for Requirements Engineering. In *Managing Requirements Knowledge*, 2013, pp. 23-47.

[2] E. Onyeka, A. Varde, O. Daramola. Common Sense Knowledge, Ontology and Text Mining for Implicit Requirements. *CSREA DMIN 2016*, pp.146-152.

[3] N. Tandon, G. de Melo, F. Suchanek, G. Weikum. WebChild: Harvesting and organizing commonsense knowledge from the web. *ACM WSDM 2014*, pp. 523-532.

[4] E. Onyeka. A Process Framework for Managing Implicit Requirements Using Analogy-Based Reasoning. *PhD Thesis, Covenant Univ. 2017.*

[5] U. Shah, D. Jinwala. Resolving ambiguities in natural language software requirements *ACM SIGSOFT J. Software Eng. Notes 2015*, 40(5):1–7.

[6] A. Umber, I. Bajwa, M. Naeem. NL-based automated software requirements elicitation and specification. *Advances in Computing and Communications 2011*, pp. 30-39.

[7] Spanoudakis, G. (1996). Analogical reuse of requirements specifications: A computational model. *Applied Artificial Intelligence*, 10(4), 281-305.

[8] Tandon, N., Varde, A., de Melo, G, Commonsense Knowledge in Machine Intelligence, *ACM SIGMOD Record 2017*, 46(4): 49-52.

[9] Lenat, D. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM* 1995, 38(11): 32-38.

[10] Miller, G. WordNet: A Lexical Database for English. *Communications of the ACM* 1995, 38(11): 39-41.

[11] Pandey, A., Puri, M. Varde, A. Object Detection with Neural Models, Deep Learning and Common Sense to Aid Smart Mobility. *IEEE ICTAI 2018*, pp. 859-863.

[12] Persaud, P., Varde, A., Robila, S. Enhancing Autonomous Vehicles with Commonsense. *IEEE ICTAI 2017*, pp. 1008-1012.

[13] Puri, M., Varde, A., Dong, B. Pragmatics and Semantics to Connect Specific Local Laws with Public Reactions, *IEEE BigData 2018*, pp. 5433-5435.

[14] Puri, M., Du, X, Varde, A., de Melo, G. Mapping Ordinances and Tweets using Smart City Characteristics to Aid Opinion Mining. *WWW Companion Vol.*, pp. 1721-1728.