Data Anonymization for Requirements Quality Analysis: a Reproducible Automatic Error Detection Task

Juyeon Kang[†*] **Jungyeul Park**[‡*]

[†] PROMETIL, 52 rue Jacques Babinet. 31100 Toulouse, France.
j.kang@prometil.com
[‡] CONJECTO, 74 rue de Paris. 35000 Rennes, France.
jungyeul@conjecto.com

Abstract

In this work, we aim at identifying potential problems of ambiguity, completeness, conformity, singularity and readability in system and software requirements specifications. Those problems arise particularly when they are written in a natural language. While we describe them from a linguistic point of view, the business impacts of each potential error are also considered in system engineering context. We investigate and explore error patterns for requirements quality analysis by manually analyzing the corpus. This analysis is based on the requirements grammar that we developed in our previous work. In addition, this paper extends our previous work in a two-fold way: (1) we increase more than twice the number of evaluation data (1K sentences) through a manual verification process, and (2) we anonymize all sensible and confidential entities in evaluation data to make our data publicly available. We also provide the baseline system using conditional random fields for requirements quality analysis, and we obtain 79.47% for the F_1 score on proposed evaluation data.
Keywords: Requirements quality analysis (ReQA), error detection in ReQA, data anonymization, reproducible task

1. Introduction

Among technical documents, requirements are a central issue since they must comply with a high number of constraints of e.g. readability, lack of ambiguity and implicit data, feasibility, relevance, traceability, conformity and overall cohesion and coherence (Firesmith, 2003; Alred et al., 2011). For example, *Ambient pressure shall be permanently maintained* relies too much on the operator's knowledge and practice: what pressure should be maintained to be ambient and what to do in case of interruption? A wrong interpretation may lead to accidents and damages.

There exists different types of references for guiding the writing of high quality requirements. Among them, we can cite:

- the standards of IEEE (ISO/IEC/IEEE29148:2011),
- ARP4754A (Aerospace Recommended Practice),
- the recommendations of INCOSE (Guide for Writing Requirements), and
- IREB (International Requirements Engineering Board)

We can also refer to the principles of controlled natural languages, mainly, those defined in ASD-STE100 (Simplified Technical English by Aerospace and Defense Industries Association of Europe). Those documents show that requirements must be non-ambiguous consistent, correct, complete, verifiable, singular, readable, feasible and traceable. However, when writing or revising a set of requirements, it is particularly challenging to make sure that texts read easily and are unambiguous for any domain actor (Weiss, 1990; Grady, 2013). There are several factors like overload, time missing, novices and specific domain knowledge needed. Tools controlling the authoring quality of requirements can be useful for automatically proofreading large quantities of requirements. Since we presented the very first linguistic model for requirements quality analysis (ReQA) (Kang and Park, 2016), there are many requests for releasing data. While we build training data from the public domain corpus (Baroni et al., 2009) using an automatic method to boost the overall learning ability, we extract sentences from the actual documents for requirements quality and annotate manually error labels for evaluation data. Most documents of requirements often contain restricted information, and documents that we used for evaluation data are also confidential in which we cannot directly release them. Therefore, in this paper, we perform the data anonymization process for ReQA evaluation data, and we release them for a reproducible automatic error detection task. Training and anonymized evaluation data are available at `https://github.com/jungyeul/rqa`.

2. Errors in ReQA

The requirements grammar and error patterns have been developed in Kang and Park (2016) to generate different models of requirements quality checking system. It was the first work to show the quantitative approach for ReQA including building training and evaluation data. The objective of our system is to verify the conformity of natural language requirements regarding to the criteria for high quality requirements as defined in the IEEE standards and the recommendations of INCOSE. Among the criteria for writing good requirements, we focus on the following five constraints: non-ambiguity, completeness, readability, conformity and singularity. These constraints can be checked by lexical and syntactic rules. For example, the Completeness concerns the use of passive voice and some incomplete expressions like *TBC* (To Be Confirmed) and *etc*. And the Singularity is not respected when a requirement contains too many combinators *and*. Each constraint is so checked

[*] Both authors contributed equally to this work.

when a requirement contains one of the lexical or syntactic errors as listed in Figure 1.

3. Anonymization

Among technical documents, Software and system Requirements Specifications (SRS) contain highly confidential data, so that the information security must be guaranteed. For example, a SRS of a vehicle includes software, hardware and networking subsystems that make up the total system. These documents specify key performance parameters (such as operations speed, response time, availability, portability and maintainability), functional capabilities, data structure and elements, safety, constraints, etc. The loss of this information can lead to significant financial losses and reputational damage. Thus, the accurate data anonymization allows to preserve the confidentiality of requirements documents and makes them useful as a linguistic corpus for a Natural Language Processing (NLP) task like automatic error detection.

Different approaches for the data anonymization were studied and applied mainly in preserving privacy from social media and health information from medical data. In social network sites (SNS), web pages are designed for human interaction like sharing opinions, experiences and feeling between unknown users of any domain. Massive data flow in SNS contains personal information (birthday, email address, social relationship, friendship, etc.) which should be protected. For the purpose of preserving such a sensitive data, some studies focus on anonymization techniques of those data. Among anonymization techniques and algorithms developed for the privacy protection on relational data, Fredj et al. (2015) describes an approach based on the generalization in order that the illustration of different generalization algorithms helps data publishers selecting an adequate technique for each data. Zhou et al. (2008) presents privacy information modeling methods along with the state-of-art anonymization approaches: clustering-based approach and graph modification approach. In clinical research, the health data anonymization is required for the protection of patient records. Dernoncourt et al. (2017) proposes a de-identification system of health information, referred to as protected health information (PHI), namely age, contact, date, ID, location, name, profession. Based on Artificial Neural Networks (ANN) and Conditional Random Fields (CRF) models, they obtained F_1-scores of 99.229% and 99.023%, respectively, on MIMIC de-identification datasets, showing that the ANN model outperforms the CRF model on all types of datasets on which the system is tested. In this work, we propose to anonymize sensitive elements that may result in the problem of confidentiality in using requirements documents. It is first necessary to identify confidential information. For that purpose, we manually analyzed 15 SRSs with more than 4000 requirements of different types and domains coming from several companies and organizations.

Figure 2 shows the most commonly appearing sensitive data in requirements documents: the named entities, quantitative values, parametric values and references. Named entities expose domain specific terms and acronyms proper to a specific project. Domain specific sensitive terms, due to their specificity, provide more information than common terms. These terms have significant meaning in a specific domain. For example, *ice condition* can be referred to as a meteorological term in a common sense, but in terms of the aviation safety, *ICE CONDITION* indicates a specific status waiting an action. An acronym ACU can be referred to as *Australian Catholic University* or *A confirmer ultérieusement* in a French SRS, but in a more specific context, ACU (Air Control Unit) is related to air flow controller which gives information on its used domain. Quantitative values disclose direct information about key performance parameters while parametric values do not directly expose parameters. However, it is possible that the latter may be indicative of different parameters. Concerning the references, some of standards and norms might be public but many of requirements also refer to internal guidelines. Thus, it will be reassuring to take them into account. Once identified the most sensitive elements to anonymize, we manually made up a dictionary of entities including domain specific terms, acronyms and parametric values. Based on this predefined dictionary, we replace original data by randomizing the characters of entities and numeric values. The dictionary of entities and anonymized data that is used in this work as evaluation data set was validated by an expert in Requirements Engineering in terms of data security and utility. During our survey, we could not find contributions applied to the data anonymization of requirements, mostly because the requirements are confidential texts which are not intended to be publicly released. Therefore, this paper focuses on the identification of sensitive and confidential information from the requirements documents.

4. ReQA Data

4.1. Training data

We already detailed how to build training data by using part-of-speech (POS) tagging and syntactic parsing in Kang and Park (2016). Since we use the automatic method to build the training data, we minimize the POS label errors and parsing results by introducing the filtering method. We use the consensus results $\hat{\mathcal{D}}$ by intersection between two results using $\mathcal{D}(\mathcal{M}_1) \cap \mathcal{D}(\mathcal{M}_2)$ where $\mathcal{D}$ is raw text data, $\mathcal{M}_i$ is a learning algorithm to annotate raw text data, and $\hat{\mathcal{D}}$ is filtered annotated data. For POS tagging, we use Hidden Markov Model (HMM) and CRF learned labeling models as described in Brants (2000) and Lavergne et al. (2010). For dependency syntactic parsing, we use two pre-trained dependency parsing models of MaltParser (Nivre et al., 2006), which uses Support vector machines (SVMs) with a polynomial kernel and linear SVMs. For raw text data, we use the first part of ukWaC, one of the WaCky corpora (Baroni et al., 2009). After consensus filtering for POS tagging and syntactic parsing, we assign five types of errors: non-ambiguity, conformity, completeness, singularity and readability. Figure 3, presented in Kang and Park (2016), shows an example sentence from our final training data. In this figure, first, *analytes or investigations* represents the ambiguity error because of *or* as explained in the Figure 1. Second, *shall be selected* is annotated as completeness error because the information about who realize the required action is not specified. Third, *their clinical relevance* also has

Singularity	Combinators: (i) `X and X'` where POS (part-of-speech) labels (or phrase type) of X and X' are same. X= verb (infinitive form), verb phrase, noun, noun phrase, adjective, value followed by a unit of measurement
Completeness	Passive construction: (i) `modal(shall) be {AdvP} Verb (Action, pp)`: *shall be used*, *shall be properly used* Pronouns: (i) `Pron (possessive) Noun`: *their application*; (ii) `Pron (possessive) NP`: *their proper development*; (iii) `Pron (demonstrative) modal(shall)`: *this shall*, *these shall*
Conformity	Negations: (i) `modal Neg`: *shall not* Modals: (i) `modal {AdvP} Verb(Action, inf)`: *would implement*, *should correctly implement*
Readability	Lexical items: quantifiers (e.g. *each*, *some*, *all*) and acronyms (e.g. *ACR*, *CPU*)
Ambiguity	Combinators: (i) `X or X'` where POS labels (or phrase types) of X and X' are same. X= verb (infinitive form), verb phrase, noun, noun phrase, adjective, value followed by a unit of measurement. Lexical items: confusing terms, vague adjectives, adverbs

Figure 1: Five constraints and their errors patterns

Named entities	Domain specific terms: *ICE CONDITION, Torque motor, Pre-cooler* Acronyms: *EMCU, MEK, AMMC, JSC, ECU*
Parametric values	*ASTM-G-3769, 5V_UPS, @TM_REACH_DEFAULT*
Quantitative values	any numeric symbols followed or preceded by units of measurement
References	Standards/Norms: *MIL-STD-753C, MT1, DO-160F* Requirements ID: *MCS-QQ-P-47*

Figure 2: Sensitive elements in requirements documents

The	DT	O
analytes	NNS	B-AMBI
or	CC	I-AMBI
investigations	NNS	I-AMBI
covered	VBN	O
by	IN	O
the	DT	O
Scheme	NNP	O
shall	MD	B-COMP
be	VB	I-COMP
selected	VBN	I-COMP
on	IN	O
the	DT	O
basis	NN	O
of	IN	O
their	PRP$	B-COMP
clinical	JJ	I-COMP
relevance	NN	I-COMP
.	.	O

Figure 3: An example sentence from training data.

	sentences	tokens
train	35,826	590,886
dev	3,980	64,769
eval	988	23,076

Table 1: Number of tokens and sentences

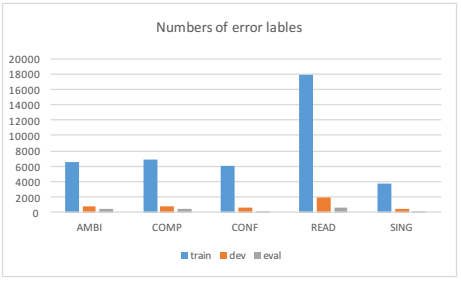

Figure 4: Number of annotated error labels

the completeness error because of the possessive pronoun *their*. It probably refers to one of the following antecedents: *analystes*, *investigations*, *the Scheme* but we need extra information to correctly identify the reference of *their*.

According to the standards IEEE (29148:2011), a requirement should be concise and a single sentence. In this paper, we add a *new* constraint $10 \leq n \leq 80$ where n is the length of sentences. Note that Kang and Park (2016) doesn't have the length constraint, and it yields the smaller data size. Then, we split 90-10 ratio for training and development data sets. Table 1 shows the number of tokens and sentences, and Figure 4 shows the number of annotated error labels in the current data sets.

4.2. Evaluation data

Among the SRSs documents that we have mentioned in §3., we extracted requirements which well represent four types of sensitive elements (Figure 2). Previously, evaluation data was composed of 319 technical requirements (481 sentences with 10,324 tokens) (Kang and Park, 2016). We extend the previous evaluation data to almost 1,000 sentences for a new evaluation data set, and they are all anonymized for data distribution as described in §3.. Based on what we define anonymization entities, we manually make up a dictionary of anonymization entities (over 270 entities) extracted from the original requirements of the extended evaluation data, and we randomize their alphanumeric characters. Figure 5 shows an example sentence of anonymized evaluation data. We add `#annonymized` for anonymized entities. `JKS`, `T2` and `E99/J04DR` are origi-

The	DT	O	
JKS	NN	O	#annonymized
internal	JJ	O	
transducer	NN	O	
T2	NN	O	#annonymized
shall	MD	O	
correspond	VB	O	
to	TO	O	
a	DT	B-READ	
E99/J04DR	NN	O	#annonymized
in	IN	O	
terms	NNS	O	
of	IN	O	
material	NN	O	
and	CC	O	
pin	NN	O	
configuration	NN	O	
.	.	O	

Figure 5: An example sentence from anonymized evaluation data.

nally for the name of the transducer and the connector, and they are completely anonymized.

5. Experiments and Results

We use CRFs (Lafferty et al., 2001) for training and testing with proposed data sets.[1] ± 2 word/POS window context and a bi-gram word/POS model are used as a feature set.[2] We use 0.1 for L2-regularization. We obtain 79.49% F_1-score on the new evaluation data set. Table 2 shows the detailed results for each error label. READ error labels are entirely based on lexical information and we correctly annotate over 95% of them because we have enough lexical information in training data. CONF error labels show only about 30% of precision because even though the expected modals as erroneous should have been detected exclusively in the main clauses, many of them were identified in the subordinated clauses where their use is allowed. AMBI and SING error labels for the combinator error pattern are usually required parsing results. While we used parsing results for building training data, we didn't consider these results for training the models. This is one of the main reasons that AMBI error labels have a relatively low recall. However, dependency information is difficult to be integrated in the sequence labeling model with dependency distance. Moreover, dependency parsing results are not often correct for conjunction marks such as *or* and *and*, which we use for the combinator error pattern.

6. Conclusion and Future Perspectives

We have presented linguistic models and a new anonymized evaluation data set for ReQA. A tool helping to improve the requirements authoring quality allows to reduce multiple proofreading steps which are time-consuming and costly but crucial in the whole life cycle of the Requirements Engineering. The accuracy of this kind of tool is obviously very important as technical authors (users) can reject to use them once they generate false positives of more than 20%. To reduce the rate of false positives, the model that we developed is based on the error patterns manually identified in the linguistic framework of the requirements grammar. In this paper, we can enrich the error patterns depending on the lexical information by adding more lexical items into our model. Additionally, the five constraints and the corresponding errors patterns do not cover all of the potential errors. It is necessary to revise and complete them in order to detect other error types: (1) detection of overspecified elements (design/solution parts – how the system realizes the required action – included in the requirements) for the Singularity, (2) detection of grammatical errors (e.g. ditransitive verbs missing one of arguments like *the system shall send the received configuration*) for the Completeness. There are also another types of constraints more ambitious such as the problem of consistency and of redundancy between requirements or sets of requirements. For those errors, we need to consider contextual information over a requirement sentence and to understand semantic meaning of the requirements and the relation between them. These are a great challenge for ReQA and we leave them as a future work of this paper.

	prec.(%)	recall(%)	F_1(%)	num.
AMBI	40.31	70.85	51.38	392
COMP	75.68	95.61	84.49	403
CONF	32.56	97.67	48.84	129
READ	91.57	98.80	95.05	629
SING	71.20	90.67	79.77	191
(all)	69.78	92.34	**79.49**	

Table 2: Experiment results: precision, recall, F_1-scores, and the number of entities.

[1] We use Wapiti (Lavergne et al., 2010) for training and evaluation, available at `https://wapiti.limsi.fr`.

[2] A trained model and training parameters are also available at `https://github.com/jungyeul/rqa`.

7. Bibliographical References

Alred, G. J., Brusaw, C. T., and Oliu, W. E. (2011). *Handbook of Technical Writing, Tenth Edition*. St. Martin's Press; 10 edition, New York.

Baroni, M., Bernardini, S., Ferraresi, A., and Zanchetta, E. (2009). The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Brants, T. (2000). TnT – A Statistical Part-of-Speech Tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, Seattle, Washington, USA. Association for Computational Linguistics.

Dernoncourt, F., Lee, J. Y., Uzuner, O., and Szolovits, P. (2017). De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606.

Firesmith, D. (2003). Specifying Good Requirements. *Journal of Object Technology*, 2(4):77–87.

Fredj, F. B., Lammari, N., and Comyn-Wattiau, I. (2015). Abstracting Anonymization Techniques: A Prerequisite

for Selecting a Generalization Algorithm. *Procedia Computer Science*, 60:206–215.

Grady, J. O. (2013). *System Requirements Analysis*. Elsevier; 2 edition.

Kang, J. and Park, J. (2016). Generating a Linguistic Model for Requirement Quality Analysis. In *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Posters (PACLIC 30)*, pages 439–447, Seoul, Korea.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lavergne, T., Cappé, O., and Yvon, F. (2010). Practical Very Large Scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 504–513, Uppsala, Sweden. Association for Computational Linguistics.

Nivre, J., Hall, J., and Nilsson, J. (2006). MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*.

Weiss, E. H. (1990). *100 Writing Remedies: Practical Exercises for Technical Writing*. Greenwood.

Zhou, B., Pei, J., and Luk, W. (2008). A Brief Survey on Anonymization Techniques for Privacy Preserving Publishing of Social Network Data. *ACM SIGKDD Explorations Newsletter*, 10(2):12–22, 12.