Summarization of Elicitation Conversations to Locate Requirements-Relevant Information

Tjerk Spijkman[1,2] [0000−0003−2726−3065], Xavier de Bondt[1,2], Fabiano Dalpiaz[1] [0000−0003−4480−3887], and Sjaak Brinkkemper[1] [0000−0002−2977−8911]

[1] Dept. of Information and Computing Sciences, Utrecht University, The Netherlands
[2] fizor., Utrecht, The Netherlands
{tjerk.spijkman, xavier.de.bondt}@fizor.com
{f.dalpiaz, s.brinkkemper}@uu.nl

Abstract. [Context and motivation] Conversations around requirements, such as interviews and workshops, are a key activity of requirements elicitation, and play a significant role in the creation of requirements specifications. **[Question / problem]** While these conversations contain a wealth of knowledge, requirements engineers use them mainly through note-taking during the conversation and by recalling the information from their memory. There is potential for supporting practitioners by retrieving important information from the recordings of these conversations. **[Principal ideas / results]** Although transcriptions can be automatically generated with good accuracy, they often contain excessive text to be efficiently used for processing requirements elicitation sessions. Thus, we observed a need to transform these datasets into a useful format for requirements engineers to analyze. **[Contribution]** We present REConSum, a prototype that utilizes Natural Language Processing (NLP) to summarize requirements conversations. REConSum takes as input a transcribed conversation, and it filters the speaker turns by keeping only those that include a question and that are expected to contain, or to be answered with, requirements-relevant information. In addition to presenting REConSum, we experiment with different algorithms to assess the most effective combination.

Keywords: Requirements Elicitation · Natural Language Processing · Conversational RE · Requirements-Relevant Information

1 Introduction

Requirements elicitation concerns the activities of seeking, uncovering, acquiring, and elaborating requirements [38]. This information is often gathered through conversational activities in which a requirements engineer (or analyst) works with system stakeholders to get an understanding of the goals and design of the system [9]. According to the NaPiRE survey [35], interviews and facilitated sessions such as workshops are the most frequently used elicitation techniques: 73% and 67% of the respondents state to be using them, respectively.

Researchers have studied requirements conversations, notably interviews, and found that note-taking is a useful activity [16] for the early detection of common problems such as ambiguity [33]. However, these conversations can range from a few hours to multiple days [2,28], thereby making it not only likely for the analyst to miss out on certain information, but also cognitively demanding as they would need to focus both on the note-taking and on keeping a natural flow.

The recordings of requirements conversations contain valuable information that can easily be lost in the overall picture of the elicitation. While creating and investigating transcriptions can be time consuming, the increasing remote work – including the online conduction of interviews and workshops – offers the opportunity to use the capability of modern online meeting tools like Microsoft Teams and Zoom Meetings to generate transcriptions that consistently improve their precision through neural network approaches [3].

Although manual reviews are possible for short conversations and they are a useful educational tool [30], we argue that analysts need to be supported in the analysis of longer real-life conversations, and that Natural Language Processing (NLP) can be fruitfully used to such an extent.

In this paper, we propose ReConSum (Requirements Elicitation Conversations Summarizer), a NLP prototype tool that can assist practitioners and researchers in processing elicitation conversations by summarizing the transcriptions and extracting requirements-relevant information. We utilize the *Question & Answer (Q&A)* structure prevalent in conversations, as discussed in Sect. 2. ReConSum retains only relevant questions and their answers through *extractive summarization* [12], thereby making transcripts easier to review, as practitioners would see a short version of the original conversation transcript. The outputs of ReConSum are meant to be used in a front-end to enable exploration of such conversations, as per the mockup of Fig. 1.

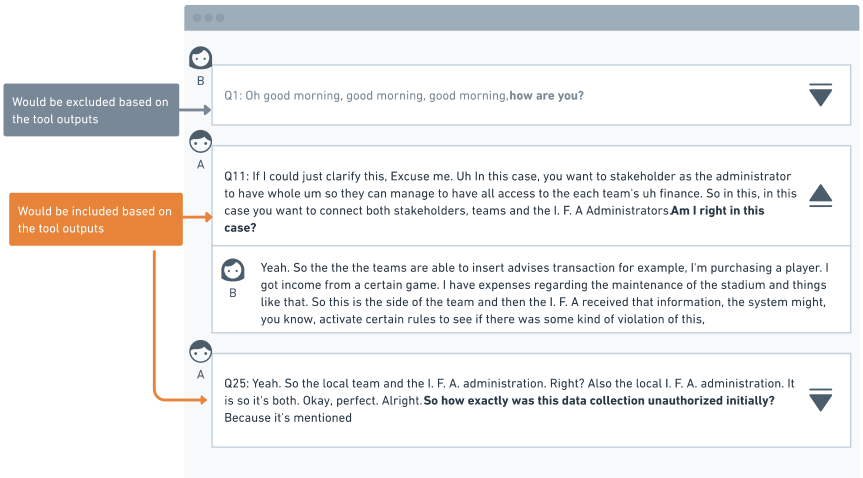

Fig. 1: Mockup visualization of the outputs of ReConSum. In this example, questions Q1–Q10 and Q12–Q24 are hidden as they are expected to be irrelevant.

To effectively summarize a document, one needs to gain a deep understanding of the document to gather the relevant information. In our context, this amounts to identifying and extracting *requirements-relevant information* from the transcript of a requirements conversation. We build on previous research [29,28]: through data investigation and experimentation with a focus group of RE students and practitioners, we gained a first understanding of what requirements-relevant information exists in requirements conversations. Based on these premises, we define the following research questions:

MRQ. How can we identify requirements-relevant information in a transcript of a requirements elicitation conversation?
RQ1. How can we define requirements relevance in a transcript?
RQ2. How to design an automated approach for locating requirements-relevant information?

The rest of this paper is structured as follows. In Sect. 2, we discuss background and related work. Sect. 3 outlines the research method and describes RECONSUM. We report on a validation of the approach in Sect. 4, and finally, we present a discussion and future works in Sect. 5.

2 Background

Conversation Structures. There are many different types of conversations, including small talk, troubles telling, and elicitation conversations [18]. Conversation analysis (CA) is the systematic analysis of the talk produced in everyday situations of human interaction [20]. CA goes beyond the scope of the spoken words and it includes video recordings of the workplace, or the onscreen activities for a conversation between people playing a game. Mondada [23] states that CA can utilize interviews not as a methodological resource for gathering information, but to study how specific practitioners work. As interviews are a natural setting for RE practitioners, this fits well in the context of this research.

We focus on the textual transcripts of a conversation: a sequence of utterances (roughly, sentences) that are spoken by one of the participants. A set of contiguous utterances by the same speaker is called a speaker turn. We acknowledge that this is only a partial picture of a conversation, which excludes aspects such as the use of artifacts (e.g., whiteboards), the analysis of intonation cues and visual cues. Furthermore, the automated transcripts delivered through video conferencing tools do not adhere to the standards used in conversation analysis, as they do not identify elements such as pauses, speed/tempo of speech, and overlapping talk [19]. While these additional perspectives are part of our future work, in this paper, we focus on automatically generated transcripts, which are already a valuable resource that is generally not considered by RE researchers and practitioners.

Another topic of conversational analysis is the identification and characterization of recurrent interaction practices [26]. These consist of a set of

actions: asking, telling, requesting, inviting, complaining, etc. One of the key concepts in interaction practices is the *adjacency pair* [31], resulting from the turn-taking format that is common in conversations [34]. Adjacency pairs are based on the understanding that an utterance is related to what comes before, and what comes next. An adjacency pair is composed of two speaker turns uttered by different speakers and placed adjacently [24]. For instance, a typical type of pair is "Request for information" followed by "Informative answer".

Stolcke *et al.* [32] discuss the lack of consensus on describing discourse structure; however, they argue that dialogue acts (DAs) are a useful first level of analysis. A DA is roughly equivalent [32] to speech acts and adjacency pairs. There are, however, differences between these theories. Take two questions such as: "Did you do it?" and "What did you wear today?". While speech acts [25] consider both questions as illocutionary acts, DAs classify these into a Yes-No question and Wh-question, respectively. DAs are a method for classifying discourse data using 42 different labels (see Table 1 for some examples). In our research, we utilize libraries that enable the automated identification of DAs to perform extractive summarization.

Related Works. In our previous research, we performed an empirical study to determine the contents of one particular RE conversation common in practice: fit-gap analysis [28], which aims to distinguish between those parts of a software product that already fit the client from those gaps that needs to be addressed via configuration or customization. The understanding gained from this work was used in designing the Trace2Conv prototype tool that assists in establishing automated pre-requirements specification traceability [29].

While focused on a different type of artifact, i.e., requirements specifications, Abualhaija *et al.* [1] apply supervised machine learning to recognize and demarcate requirements in a free-form requirements specification. In their work, modal verbs are used to determine important segments of the text, and parts of their NLP pipeline inspired this work.

Another adjacent area of research is that of automated requirements classification [6], which led to the development of NLP tools that organize requirements into categories. A typical classification distinction is between functional and non-functional requirements. To achieve such a classification, Kurtanović and Maalej [22] apply supervised machine learning through support vector machines. Their research shows that POS tags, word n-grams, modal verbs and the POS tag 'cardinal number' were the most informative. Similarly, higher-level linguistic dependencies can be useful in classification, as shown by Dalpiaz *et al.* [7]. This work also observes that the performance of classifiers degrades when used on other datasets; this led to the birth of the ECSER pipeline for a rigorous evaluation of classifiers in software engineering [10].

Only a few scholars have studied requirements conversations in depth. Alvarez and Urla [2] performed a manual analysis of interview transcripts concerning the construction of an ERP system; they studied the role of stakeholders and of client stories. Ferrari *et al.* [16] conducted and analyzed 34 simulated interviews and identified four facets of ambiguity (unclarity, multiple

understanding, incorrect disambiguation and correct disambiguation); moreover, they found that ambiguity can be a cue for the elicitation of tacit knowledge. Their follow-up work [14] explores the use of voice and biofeedback to identify emotions that may represent engagement. Bano *et al.* [4] analyze interviews by novices and they build an educational framework for teaching how to conduct requirements interviews. The same authors extend their approach by including (reverse) role-playing elements [15].

3 RECONSUM: a Tool for Summarizing RE Conversations

We designed RECONSUM according to the phases of the engineering cycle by Wieringa [36], focusing on the first three phases (the design cycle) as the current prototype has not yet been applied to practical cases. The code can be found in GitHub[3] and a persistent copy is in the online appendix [27]. In this section, we discuss the problem investigation and solution design steps of Wieringa's design cycle. Sect. 4 reports on the validation step.

3.1 Problem Investigation and Solution Design Iterations

The problem addressed by this research is based on observations of the artifacts from our previous research [28,29]. We found that RE conversations contain useful information, but that their manual analysis is time consuming. This indicates an opportunity for designing intelligent tools that can reduce the necessary effort through automation.

To further explore the problem domain, we analyzed nine recordings of requirements interviews conducted by master's students at Utrecht University in a simulated setting. These recordings were split across three different cases (see [8] for details): IFA – the international football association portal, UMS – an urban mobility simulator, and HMS – a hospital management system. All interviews had a similar time frame (max. 60 minutes) and structure, thereby allowing us to consider multiple cases per domain as well as different domains.

Given the limited existing literature, the treatment design is approached in an exploratory way. We went through multiple design iterations, which had the definition of the term *requirements-relevant information* as a recurring theme. Although a seemingly simple term, we found that answering the question "what does it mean to be relevant" is hard. There is no single recipe to define if something is relevant to an analyst, as this may depend on the specific use case, e.g., authoring requirements, searching for missing requirements, tracing requirements backwards, and implementing the requirements.

Discarded designs. We initially expected to utilize the categories of requirements-relevant information that we defined in previous research [28,29]. However, we found out that these categories were too context reliant. For example, the *as-is process* is fundamental when replacing a legacy system,

[3] `https://github.com/RELabUU/REConSum`

while the *to-be process* is more important for a new system design [28]. Additionally, our collection of transcripts was heterogeneous regarding the design activity (greenfield vs. brownfield), and several sentences were hard to classify as they described the as-is by indicating something about the to-be.

We also attempted the design of a machine learning-based automated binary classification of the transcript, with *speaker turns* being either relevant or irrelevant. The main challenges with this approach, however, were the heterogeneity of the transcripts and the too limited amount of labeled data.

Extractive question-based summarization. Through further exploration of the artifacts, we found that the Q&A structure could effectively be used to produce a condensed version of the conversation while still covering most of the content. The idea was that of retaining only those questions that are (potentially) relevant, and the analyst could then further navigate the conversation by zooming in on the answers of those questions as shown in Fig. 1. This led to the design detailed in this section; an approach that first classifies the text based on its structure, focusing on the identification of a special kind of adjacency pairs: questions and answers to those questions.

3.2 Question Identification and Relevance Detection

We aim to obtain an extractive summarization of a requirements conversation consisting of only those speaker turns that include questions that are potentially relevant for requirements engineers.

Formally, let a conversation $C = (T_1, T_2, \ldots, T_n)$ be a sequence of speaker turns, where $n \in \mathcal{N}^+$ is a positive natural number. A speaker turn T_i is a sequence of utterances (roughly, sentences) that are spoken by the same speaker: given $i, m \in \mathcal{N}^+$, $T_i = (U_1, U_2, \ldots, U_m)$. An utterance U_j is a sequence of words by the same speaker: given $j, p \in \mathcal{N}^+$, $U_j = (w_j^1, \ldots, w_j^p)$. We define two functions. $\mathit{IsQuestion} : \mathcal{U} \to \{0, 1\}$ is a Boolean function that returns 1 if and only if $U \in \mathcal{U}$ is a question. $\mathit{IsRelevant} : \mathcal{U} \to \{0, 1\}$ is a Boolean function that returns 1 if and only if $U \in \mathcal{U}$ is a relevant utterance for a requirements engineer. We can now formally define our summarization function $\mathit{Summ} : \mathcal{C} \to \mathcal{C}$, where $\mathcal{C}$ is the domain of conversations, as follows:

$$\mathit{Summ}(C) = S.\ S \text{ is a sub-sequence of } C \text{ and } \forall\, T = (U_1, \ldots, U_m) \in S,$$
$$\exists k \in [1, m].\ \mathit{IsQuestion}(U_k) \wedge \mathit{IsRelevant}(U_k)$$

RECONSUM Process. Fig. 2 provides an overview of how RECONSUM implements the summarization function *Summ*. It takes a requirements interview transcript as an input, and first determines which speaker turns contain a question. This is done utilizing Part-of-Speech tagging and/or Dialogue Act classification. Then, RECONSUM determines if these questions are relevant by assessing whether they contain domain-specific terms; our assumption is that the presence of those terms is an indicator of relevance.

We determine whether a speaker turn includes domain-specific terms by calculating Term Frequency–Inverse Document Frequency technique

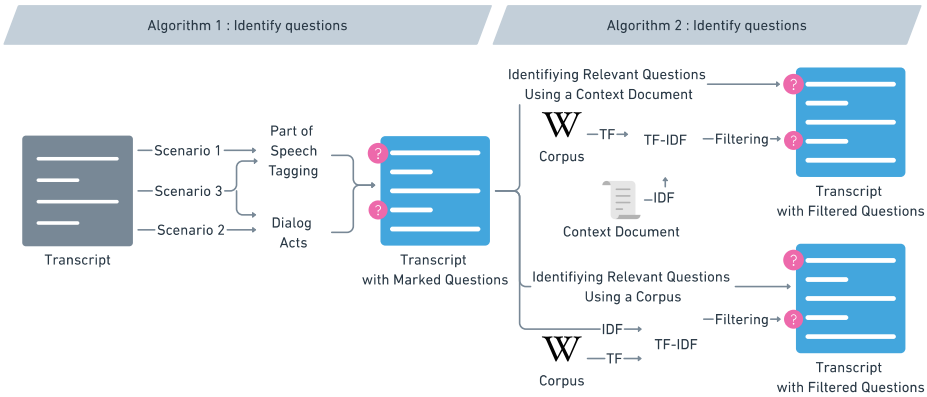

Fig. 2: A process flow overview of ReConSum.

(TF–IDF). We first compute the Inverse Document Frequency (IDF) either of the transcript itself (bottom-right scenario in Fig. 2) or of a context document (top-right scenario in Fig. 2). We then calculate Term Frequency (TF) of a Wikipedia corpus, which we take as a general-purpose corpus where the distribution of the terms is not expected to reflect the specificity of the conversation domain. Then, ReConSum retains only those speaker turns that both include a question as well as words with a high TF–IDF score, indicating that these terms are much more frequent in the domain than in Wikipedia.

Algorithm 1: Identify questions. The first stage of ReConSum is to identify the questions in a transcript through (i) a deep learning classifier based on dialogue acts, (ii) the occurrence of sequences of Part-of-Speech (POS) tags, or (iii) either of the previous. The dialogue acts approach assigns a dialogue act to each sentence in the speaker turn, while the approach based on POS tags assigns these tags to parts of each sentence. These POS tags were taken from the Penn Treebank POS Tagset[4], which contains two clause-level tags that can indicate questions [5]: `SBARQ` and `SQ`. These tags indicate four types of questions: *wh*-questions, *yes-no*-questions, *tag*-questions, and *choice*-questions.

Our dialogue act-based approach relies on the off-the-shelf classifier `DialogTag`[5], which uses a neural architecture based on BERT [11] to assign a dialogue act to a sentence. `DialogTag` uses a subset of the Switchboard-1 corpus; the latter was created using 2,400 telephone conversations, with conversations among 543 speakers on 70 topics [21]. Our implementation labels as question those speaker turns that include a sentence that denotes one of the question types that `DialogTag` identifies; frequent examples are in Table 1.

We also propose a third approach that aims at supporting those scenarios where recall is more important than precision: we execute both of the previous approaches and retain the speaker turns if they are included in either approach.

[4] Our implementation is inspired by that of `https://github.com/garcia2015/NLP_QuestionDetector`

[5] `https://github.com/bhavitvyamalik/DialogTag`

Table 1: Examples of question types, based on dialogue act classification, that are used by function DialogueActs.

Tag	Example
Yes-No-Question	*Is there already some data that can be gathered from the existing systems that can already be put in the new one or not?*
Wh-Question	*I'm gonna ask you, how long does it take for that person to analyze the situation and uh monitor a certain road or urban traffic situations?*
Declarative Yes-No-Question	*So it would be a manual change, not a new iteration of the automated schedule.*
Backchannel in question form	*Um, this should also be made available I imagine, during a match for instance, the score of the match should be updated immediately once it's changed. Right?*
Open-Question	*What do you mean with 'local' I.F.A.?*
Rhetorical-Questions	*(...) We have done it in the in the other city the other year. So why shouldn't it work now?*
Or-Clause	*So you think there should be the same rights for every system user? Or do you think that one user should have less rights capability?*
Tag-Question	*Right?*

Algorithm 1 Identify Questions

Input: C a set of speaker turns,
Output: T the set of speaker turns, with the questions marked

function DialogueActs(C)
  for all $sent \in C$ **do**
    $T[sent] \leftarrow sent$
    $T[sent]_{question} \leftarrow$ `False`
    for all $tag \in$ DialogTag.Dialogue_Acts(Tokenize($sent$)) **do**
      if `ANY`({`-Question`, `Or-Clause`}) $\in tag$ **then**
        $T[sent]_{question} \leftarrow$ `True`
  return T

function Part-of-Speech-Tags(C)
  for all $sent \in C$ **do**
    $T[sent] \leftarrow sent$
    $T[sent]_{question} \leftarrow$ `False`
    for all $subtree \in$ NLP_Annotate($sent$) **do**
      if $subtree.POS\_tag \in$ {`SBARQ`, `SQ`} **then**
        $T[sent]_{question} \leftarrow$ `True`
  return T

function Combined(C)
  $T_1 \leftarrow$ DialogueActs(C)
  $T_2 \leftarrow$ Part-of-Speech-Tags(C)
  for $i = 0;\ i < |C|;\ i{+}{+}$ **do**
    $T[sent] \leftarrow C[i]$
    $T[sent]_{question} \leftarrow T_1[i]_{question} \vee T_2[i]_{question}$
  return T

The first step of ReConSum is detailed in Algorithm 1. An input set C of speaker turns is turned into a version T where the speaker turns with a question are marked. The DialogueActs function loops through each

sentence in C, retrieves the dialogue acts that apply to that sentence through the `DialogTag` BERT-based classifier, and determines if the sentence contains one of the dialogue acts that indicate a question. Similarly, the PART-OF-SPEECH-TAGS function generates the POS trees of a sentence, and explores each of them to see if it contains a question indication (`SBARQ` or `SQ`). The combined approach (function COMBINED) returns those speaker turns that are identified as questions by at least one of the other two functions.

Algorithm 2 Categorize Relevant Questions

Input: T a set of speaker turns, with the questions marked,
F a file to compare the relevance to; either a context document or the conversation transcript,
num_words the number of unfiltered words you would like to categorize the questions on
(set to 60 in our experiments based on empirical testing),
Output: the set of speaker turns, with the questions and their relevance marked

function CREATEWORDLIST(F, num_words)
  $IDF \leftarrow$ LOAD_WIKI_TF
  $File \leftarrow$ PREPROCESS_FILE(F)
  $Words \leftarrow$ CALCULATE_TF-IDF(IDF, $File$)
  $Word\_List \leftarrow$ TAKEFIRSTN(SORT($Words$), num_words)
  $Word\_List \leftarrow$ STEM($w \in Word\_List \mid w \notin stop\_words$)
  return $Word\_List$

function FILTERQUESTIONS(T, F)
  $Word\_List \leftarrow$ CREATEWORDLIST(F, 60)
  for all $sent \in T$ **do**
    $T[sent]_{relevant} \leftarrow$ `False`
    if $T[sent]_{question}$ **then**
      for all $word \in$ SPLIT_SENTENCE($sent$) **do**
        if STEM($word$) $\in Word\_List$ **then**
          $T[sent]_{relevant} \leftarrow$ `False`
  return T

Algorithm 2: Categorize relevant questions. In the second stage (right-hand side of Fig. 2), RECONSUM looks for *relevant* questions based on the outputs of the first stage. RECONSUM implements two approaches, both reliant on TF–IDF. In both cases, TF is calculated on the basis of a Wikipedia dataset [17]. In the first approach, we calculate IDF on a document that describes the application context (a *context document*), which provides us with words that can indicate the relevance of a question. Based on TF–IDF, we retain only questions that include words with a high TF–IDF value. In our second approach, we follow the same process, but IDF is calculated on the transcript without the need to use a context document.

This functionality is described in Algorithm 2, which includes two functions. The first one (CREATEWORDLIST) takes as input (i) the file F we use to calculate IDF: either the transcript or the context document, and (ii) an integer that indicates the number of unfiltered words to categorize the outputs. After loading the Wikipedia Term-Frequency and processing the file (F), the TF–IDF scores can be calculated. After that, we sort on these scores and take the number of unfiltered words that we specified. Finally, we remove stopwords, and we stem

the remaining words. The second function (FilterQuestions) takes as inputs the set of speaker turns, with the questions marked, from Algorithm 1 and the same file F used by CreateWordList. It first creates a list of domain-specific words by calling CreateWordList, and then marks the questions that contain one of these words as relevant, thus returning a set of speaker turns where the questions are marked and have their relevance indicated.

4 Evaluation

After explaining the design of our evaluation and golden standard in Sect. 4.1, we present qualitative and quantitative results in Sect. 4.2.

4.1 Designing the Golden Standard

With the aim of measuring the performance of REConSum in identifying only the relevant questions in a conversation, we set off to design a golden standard. To do so, we performed a number of design iterations that were meant to define the instrument through which the golden standard could be created. The goal was to have this standard created by people who are not the authors of this paper. To this end, the tagging was facilitated through a survey.

A key decision was establishing what context (how many speaker turns) should be shown for each question, as we expected it would be difficult to rate relevance without that information. We eventually decided to include the speaker turn that includes the question, the previous turn, and the next one.

Table 2: Categorization of requirements-relevant information in the tagging

Functional requirement	The speaker turn refers to functionality that the software system has to exhibit. For example, register users, schedule events, calculate something or allow messaging.
Non functional requirement	Software quality or non-functional requirement. The speaker turn refers to qualities that the system should provide while delivering its functionalities, e.g., speed, security, capacity, compatibility, reliability, usability, portability.
System users	The speaker turn mentions the users of the system, or other stakeholders that do not use the system.
Current process understanding	The speaker turn contains information about the current process or system as-is, including current problems that the interviewee is facing.
Within or outside of the scope	The speaker turn explicitly contains a discussion of elements that should be in the system to-be or not. These define the boundaries of the system's scope.
No requirements-relevant information	The speaker turn does not contain any relevant information.

Another challenge concerned deciding whether a segment was relevant. To facilitate this, we defined a categorization of relevance inspired by our earlier work [28], as shown in Table 2. Sometimes the question itself was not relevant,

yet the surrounding text was. Therefore, the taggers were asked whether requirements-relevant information: (a) could be expected in the answer to the question; or (b) could be found in the speaker turn shown after the question.

The taggers were first asked to decide whether the segment included one of the relevant categories in Table 2, and, if so, they could answer the questions regarding where the relevant information was located. All taggers were provided with a tagging guide (in our online appendix alongside the source code and the results [27]), and an overview of the case being discussed.

Execution of Tagging. We created the golden standard for the nine datasets shown in Table 3 by recruiting 18 taggers: either students familiar with requirements engineering or practitioners. Two of them tagged each dataset using a Qualtrics survey: each participant was assigned a case, and they would see all of the questions in the conversation in chronological order, as in Fig. 3. They would go through the conversation one question at a time, with the option to return to the previous question. As per Table 3, the participants saw on average circa 71% of the conversation, and they could tag for relevance 58.8% of the conversation. This difference arises because the participants could not tag the speaker turn before the one where the question is located.

Table 3: Evaluation datasets. The UMS/IFA/HMS identifier refers to the case name as per Sect. 3.1. The table also shows recording length, number of speaker turns, then number (#) and percentage (%) of speaker turns (a) shown to the taggers, (b) that could be tagged, and (c) that include questions. The 'Relevant' columns characterize the gold standard defined by the taggers, and 'Agreement' shows inter-rater agreement in percentage and using Cohen's kappa.

Set	**Length**	**Speaker Turns**												**Agreement**	
		Total	**Shown**		**Taggable**				**Questions**						
							Relevant				**Relevant**				
	mm:ss		#	%	#	%	#	%	#	%	#	%		%	k
1-UMS	50:23	167	117	70.1%	95	56.9%	61	64.2%	49	29.3%	31	63.3%		54.7%	0.20
2-IFA	49:15	148	107	72.3%	85	57.4%	67	78.8%	46	31.1%	34	73.9%		58.8%	0.21
3-UMS	41:29	98	69	70.4%	56	57.1%	36	64.3%	30	30.6%	14	46.7%		60.7%	0.16
4-HMS	23:05	69	50	72.5%	41	59.4%	31	75.6%	21	30.4%	15	71.4%		90.2%	0.75
5-IFA	58:06	179	132	73.7%	105	58.7%	51	48.6%	56	31.3%	20	35.7%		76.2%	0.53
6-HMS	38:25	116	77	66.4%	64	55.2%	44	68.8%	34	29.3%	17	50.0%		60.1%	0.22
7-IFA	47:12	162	109	67.3%	91	56.2%	79	86.8%	46	28.4%	41	89.1%		59.3%	0.07
8-HMS	39:24	155	115	74.2%	98	63.2%	68	69.4%	54	34.8%	38	70.4%		89.8%	0.76
9-HMS	30:31	80	65	81.3%	55	68.8%	39	70.9%	28	35.0%	20	71.4%		92.7%	0.80
Average	49:15	130	93	71.6%	77	58.8%	53	69.0%	40	31.0%	26	63.2%		71.6%	0.41
Total		1174	841		690		476		364		230				

Although the design was meant to ensure a common understanding, the inter-rater agreement is low (the macro-average of 0.41 is at the boundary between *fair* and *moderate*). We mainly ascribe this to the fact that we did not define a clear use case, and the notion of relevance may depend on the task at hand and domain experience. In general, we identified three common types of disagreements: (i) the statement 'Do you expect the question to be answered with requirements-relevant information' was often misread as 'Does the question include . . . '; (ii)

whether yes-no answers should be considered relevant; and (iii) if the summary of a previous answer, made by the analyst, should be considered relevant. The disagreements were first manually validated by the second author to identify obvious sloppiness, and those cases were discarded. When this analysis did not resolve the disagreements, we took an inclusive approach in which a speaker turn was considered relevant if one of the taggers tagged it as such.

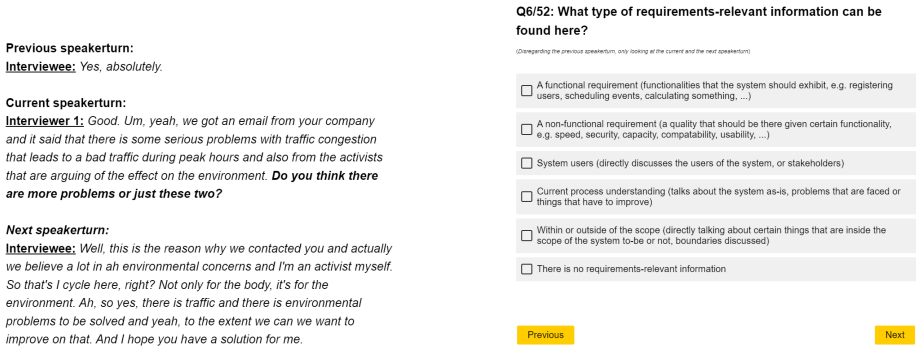

Fig. 3: Illustration of the tagging tool. On the left, the speaker turn including a question, together with the adjacent ones, are shown. On the right, the tagger selects whether and what kind of requirements-relevant information exists.

4.2 RECONSUM results

To determine the effectiveness of RECONSUM (its implementation of the extractive summarization function *Summ* in Sect. 3.2), NLP summarization task metrics could be applied, e.g., coherence, consistency, percentage of text shown [13]. Other metrics such as BLEU and ROUGE assume the existence of a reference summary, which we do not possess at this stage of our research. In this paper, we assess RECONSUM's ability of filtering, and therefore utilize standard information retrieval metrics: precision, recall, F_1-score, and accuracy. As a unit of analysis, we take speaker turns; in other words, we measure (i) if the speaker turn contains a question, and (ii) if a speaker turn with a question is relevant and should therefore be retained in the summary.

Question detection. For the first part of RECONSUM, we compare the three variants described in Algorithm 1: POS tagging, dialogue acts, and their combination. Table 4a presents a summary of the results, while the results for each dataset can be found online. The approach based on a deep learning classifier for dialogue acts leads to higher precision and higher recall than the approach based on POS tagging, perhaps thanks to the higher number of question types that it recognizes. Combining both approaches increases the number of true positives, but at the cost of increasing the false positives too.

Table 4: Performance metrics, showing macro-average and standard deviation across the nine datasets of Table 3. The best results are highlighted in green.

Approach	Precision		Recall		F_1		Accuracy	
	$\overline{x}$	σ	$\overline{x}$	σ	$\overline{x}$	σ	$\overline{x}$	σ
Dialogue acts	0.820	0.077	0.906	0.066	0.858	0.054	0.908	0.035
Part of Speech Tags	0.778	0.096	0.696	0.101	0.727	0.061	0.839	0.033
Combination	0.766	0.089	0.951	0.048	0.846	0.060	0.891	0.046

(a) Question identification task

Approach	Precision		Recall		F_1		Accuracy	
	$\overline{x}$	σ	$\overline{x}$	σ	$\overline{x}$	σ	$\overline{x}$	σ
Context Doc. - DA	0.644	0.131	0.689	0.144	0.653	0.117	0.867	0.015
Wikipedia - DA	0.642	0.112	0.653	0.076	0.640	0.074	0.861	0.011
Context Doc. - POS	0.535	0.158	0.619	0.153	0.562	0.128	0.819	0.038
Wikipedia - POS	0.532	0.148	0.625	0.126	0.564	0.110	0.816	0.040
Context Doc. - COMB	0.542	0.114	0.810	0.135	0.641	0.099	0.828	0.036
Wikipedia - COMB	0.548	0.119	0.808	0.106	0.646	0.097	0.831	0.041

(b) Question relevance task

Using the combined approach reduces the summarization rate, but at the same time it decreases the likeliness of missing requirements-relevant information.

Categorization of relevant questions. Once the questions are found in the transcript, Algorithm 2 determines if they are requirements relevant. The algorithm includes two approaches for calculating term frequency, either from the conversation itself, or from a contextual document. These can then be applied to all three approaches for question detection, leading to six combinations. The results in Table 4b show that the highest precision is obtained by combining (a) dialogue act tagging for question identification with (b) TF–IDF using a context document for relevance detection. The highest recall is obtained through the COMBINED algorithm for question detection. The latter result is not surprising, as Algorithm 2 takes as input the outputs of Algorithm 1, and the combined approach had by far the highest recall (over 95%, see Table 4a). Based on these results, we cannot identify a clear winner, as the decision depends on the relevant metrics for the use case. Precision is more important for a specification task as enrichment to note-taking, while recall is more important when searching for missed requirements.

Locating requirements-relevant information in questions. The results show that RECONSUM is able to effectively extract the questions from the conversations. In Table 3, circa 63% of these questions were tagged as relevant by the taggers. The tagging results, however, indicate a higher relevance for all taggable items (questions, plus the following speaker turn) than just the questions: 69%. Thus, the answers contain more requirements-relevant information than the questions, thereby indicating their importance for users when exploring a conversation. If we would include those answers in the summary, we would increase relevance by reducing the summarization rate.

5 Conclusions, Limitations, and Future Work

In this paper, we presented ReConSum as a step towards the summarization of requirements conversation transcripts, building on and extending the knowledge in the field of Conversational RE [29]. ReConSum employs NLP techniques to extract questions from a transcript and to determine their relevance. This approach was validated against an assembled golden standard, reaching an F_1 score around 65%. While just showing the questions might not contain all the necessary information for a RE practitioner, this is meant to be a starting point for further exploring parts of a transcribed conversation.

Although the definition of *requirements relevance* (**RQ1**) in conversations is not final, the discussions and creation of a golden standard provides further knowledge in this domain. The high disagreement across taggers shows that determining requirements relevance depends on the perspective of the individual tagger and on the use case at hand: why is the transcript being explored? Similarly, relevance cannot be determined in a vacuum (a single speaker turn), and we allowed taggers to read a context that includes the previous and following speaker turns (both adjacency pairs before and after the question). The survey design also defines a minimal categorization of data as presented in Table 2 which is the result of multiple design iterations.

To automate the identification of requirements-relevant information (**RQ2**), the best results (see Table 4) are obtained through a combination of dialogue acts classification and TF–IDF that allows question recognition and to determine their relevance. ReConSum provides these questions as a summary of the entire transcript that can facilitate exploration of the conversations by third parties or reviewing by part-taking practitioners.

The answers to RQ1 and RQ2 allow us to address **MRQ**: ReConSum is our initial answer for the automated identification of requirements-relevant information in a requirements conversation.

Threats to validity. The obtained results should be seen in light of the threats to validity, which we classify according to Wohlin *et al.* [37].

Conclusion validity. The comparison against the golden standard has some limitations, as it was created by one pair of taggers per conversations. This means we are not only comparing our tool to the golden standard, but also to the human performance in creating this standard. Additionally, we have used classic information retrieval metrics, but we did not employ classic summarization metrics at this stage, which require possessing a reference summary. Additionally, the relevance of the questions and answers were tagged by perceived relevance, but we did not tag the remaining speaker turns (those that do neither include a question or that constitute an answer to a question). It needs to be confirmed which speaker turns include the highest percentage of requirements-relevant information.

Internal validity. To make the tagging exercise easier for the participants, we used a non-exhaustive list of relevance categories, which might have impacted their perception of relevance for the tagged speaker turns. Additionally, while we

utilized generated transcripts, these were post-processed to remove transcription errors; this is likely to have a positive effect on the findings.

Construct validity. All the cases used in our validation and theory building consisted of interviews focused on the creation of one of three information systems. This homogeneity probably has an impact on the type of information to be found in the transcripts. Our results are based on the golden standard, but requirements relevance remains a term that is up to the interpretation and use case for reviewing the context. This means that the lack of clear boundaries for requirements relevance has an impact on our findings and conclusions.

External validity. Our validation and design relied on a set of simulated interviews conducted by students. Whether the results generalize to practical settings can only be determined by using interviews from real-world projects.

Future Works. The research leading to RECONSUM is part of *conversational RE*: "the analysis of requirements elicitation conversations aimed at identifying and extracting requirements-relevant information" [29]. We expect to support this goal by building RE tools that can reduce the effort for practitioners to review and explore the conversations for requirements-relevant information. We first discuss direct improvements for RECONSUM, followed by more general research directions concerning conversational RE.

As an additional functionality for RECONSUM, we experimented with applying different learning approaches (Machine Learning, Transfer Learning, and Zero-Shot Learning) to categorize questions similar to our tagging exercise. While these outputs were not significant due to the limited labeled data, we expect that extending the golden standard could enable an effective learning technique to classify data within the categories of requirements relevance. Also, an investigation of the options for user interaction starting from the outputs of RECONSUM is necessary to allow the use of the tool in practice. The interface shown in Fig. 1 is only an initial idea that shall be further developed.

Beyond RECONSUM, we can extend the *conversational RE* field in many ways. For instance, the generation of domain/data models from conversations could speed up development drastically especially in the low-code development domain. The field of conversation analysis offers many avenues for a rich exploration of conversations, e.g., exploiting multi-modal data that includes video footage, screensharing, whiteboard contents, and prototypes. Another open topic is to extend the analysis beyond single conversations into a more extensive approach that can be utilized throughout a project linking all conversations together and keeping track of changes in requirements over time.

Acknowledgements. We thank all the participants who acted as taggers. The use of the recorded and transcribed dataset is made possible thanks to the ethical Science-Geosciences Ethics Review Board of Utrecht University (case S-20339).

References

1. Abualhaija, S., Arora, C., Sabetzadeh, M., Briand, L.C., Traynor, M.: Automated demarcation of requirements in textual specifications: A machine learning-based

approach. Empirical Software Engineering **25**, 5454–5497 (2020)
2. Alvarez, R., Urla, J.: Tell me a good story: Using narrative analysis to examine information requirements interviews during an ERP implementation. ACM SIGMIS Database **33**(1), 38–52 (2002)
3. Archibald, M.M., Ambagtsheer, R.C., Casey, M.G., Lawless, M.: Using zoom videoconferencing for qualitative data collection: Perceptions and experiences of researchers and participants. International Journal of Qualitative Methods **18** (2019)
4. Bano, M., Zowghi, D., Ferrari, A., Spoletini, P., Donati, B.: Teaching requirements elicitation interviews: an empirical study of learning from mistakes. Requirements Engineering **24**(3), 259–289 (2019)
5. Bies, A., Ferguson, M., Katz, K., MacIntyre, R., Tredinnick, V., Kim, G., Marcinkiewicz, M.A., Schasberger, B.: Bracketing guidelines for Treebank II style Penn Treebank project. Tech. rep., University of Pennsylvania (1995)
6. Cleland-Huang, J., Settimi, R., Zou, X., Solc, P.: Automated classification of non-functional requirements. Requirements engineering **12**(2), 103–120 (2007)
7. Dalpiaz, F., Dell'Anna, D., Aydemir, F.B., Çevikol, S.: Requirements classification with interpretable machine learning and dependency parsing. In: IEEE Intl. Requirements Engineering Conference. pp. 142–152 (2019)
8. Dalpiaz, F., Gieske, P., Sturm, A.: On deriving conceptual models from user requirements: An empirical study. Information and Software Technology **131**, 106484 (2021)
9. Davis, A., Dieste, O., Hickey, A., Juristo, N., Moreno, A.M.: Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review. In: IEEE Intl. Requirements Engineering Conference. pp. 179–188 (2006)
10. Dell'Anna, D., Aydemir, F.B., Dalpiaz, F.: Evaluating classifiers in SE research: the ECSER pipeline and two replication studies. Empirical Software Engineering **28**(1), 1–40 (2023)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding (2018), `https://arxiv.org/abs/1810.04805`
12. El-Kassas, W.S., Salama, C.R., Rafea, A.A., Mohamed, H.K.: Automatic text summarization: A comprehensive survey. Expert Systems with Applications **165**, 113679 (2021)
13. Fabbri, A.R., Kryściński, W., McCann, B., Xiong, C., Socher, R., Radev, D.: Summeval: Re-evaluating summarization evaluation. Transactions of the Association for Computational Linguistics **9**, 391–409 (2021)
14. Ferrari, A., Huichapa, T., Spoletini, P., Novielli, N., Fucci, D., Girardi, D.: Using voice and biofeedback to predict user engagement during requirements interviews. arXiv:2104.02410 (2021)
15. Ferrari, A., Spoletini, P., Bano, M., Zowghi, D.: SaPeer and ReverseSaPeer: Teaching requirements elicitation interviews with role-playing and role reversal. Requirements Engineering **25**(4), 417–438 (2020)
16. Ferrari, A., Spoletini, P., Gnesi, S.: Ambiguity and tacit knowledge in requirements elicitation interviews. Requirements Engineering **21**(3), 333–355 (2016)
17. Galkin, M., Malykh, V.: Wikipedia TF-IDF Dataset release (2020). `https://doi.org/10.5281/zenodo.3631674`
18. Hakulinen, A.: Conversation types. In: D'hondt, S., Verschueren, J., Östman, J.O. (eds.) The Pragmatics of Interaction, pp. 55–65 (2009)

19. Hepburn, A., Bolden, G.B.: The conversation analytic approach to transcription. In: Stivers, T., Sidnell, J. (eds.) The handbook of conversation analysis, pp. 57–76 (2013)
20. Hutchby, I., Wooffitt, R.: Conversation Analysis: Principles, Practices and Applications. Wiley (1998)
21. John J. Godfrey, E.H.: Switchboard-1 release 2 (1993), https://doi.org/10.35111/sw3h-rw02
22. Kurtanović, Z., Maalej, W.: Automatically classifying functional and non-functional requirements using supervised machine learning. In: IEEE Intl. Requirements Engineering Conference. pp. 490–495 (2017)
23. Mondada, L.: The conversation analytic approach to data collection. In: Stivers, T., Sidnell, J. (eds.) The handbook of conversation analysis, pp. 32–56 (2013)
24. Schegloff, E.A., Sacks, H.: Opening up closings. Semiotica **8**(4), 289–327 (1973)
25. Searle, J.R., Searle, J.R.: Speech acts: An essay in the philosophy of language. Cambridge University Press (1969)
26. Sidnell, J.: Basic conversation analytic methods. In: Stivers, T., Sidnell, J. (eds.) The handbook of conversation analysis, pp. 77–99. Wiley Online Library (2013)
27. Spijkman, T., de Bondt, X., Dalpiaz, F., Brinkkemper, S.: Online appendix to Summarization of Elicitation Conversations to Locate Requirements-Relevant Information (2023), https://doi.org/10.5281/zenodo.7650324
28. Spijkman, T., Dalpiaz, F., Brinkkemper, S.: Requirements elicitation via Fit-Gap Analysis: A view through the grounded theory lens. In: International Conference on Advanced Information Systems Engineering. pp. 363–380 (2021)
29. Spijkman, T., Dalpiaz, F., Brinkkemper, S.: Back to the roots: Linking user stories to requirements elicitation conversations. In: IEEE Intl. Requirements Engineering Conference (RE@Next! track) (2022)
30. Spoletini, P., Ferrari, A., Bano, M., Zowghi, D., Gnesi, S.: Interview review: An empirical study on detecting ambiguities in requirements elicitation interviews. In: International Working Conference on Requirement Engineering: Foundation for Software Quality. pp. 101–118 (2018)
31. Stivers, T.: Sequence organization. In: Stivers, T., Sidnell, J. (eds.) The handbook of conversation analysis, pp. 191–209 (2013)
32. Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C.V., Meteer, M.: Dialogue act modeling for automatic tagging and recognition of conversational speech. Computational linguistics **26**(3), 339–373 (2000)
33. Sutcliffe, A., Sawyer, P.: Requirements elicitation: Towards the unknown unknowns. In: IEEE Intl. Requirements Engineering Conference. pp. 92–104 (2013)
34. Traum, D.R., Hinkelman, E.A.: Conversation acts in task-oriented spoken dialogue. Computational Intelligence **8**(3), 575–599 (1992)
35. Wagner, S., Fernández, D.M., Felderer, M., Vetrò, A., Kalinowski, M., Wieringa, R., Pfahl, D., Conte, T., Christiansson, M.T., Greer, D., et al.: Status quo in requirements engineering: A theory and a global family of surveys. ACM Transactions on Software Engineering and Methodology (2019)
36. Wieringa, R.J.: Design science methodology for information systems and software engineering. Springer (2014)
37. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering. Springer (2012)
38. Zowghi, D., Coulin, C.: Requirements elicitation: A survey of techniques, approaches, and tools. In: Engineering and managing software requirements, pp. 19–46. Springer (2005)