Is there Really a Need for Using NLP to Elicit Requirements? A Benchmarking Study to Assess Scalability of Manual Analysis

Eduard C. Groen
Fraunhofer IESE
Kaiserslautern, Germany
eduard.groen@iese.fraunhofer.de

Jacqueline Schowalter
University of Kaiserslautern
Kaiserslautern, Germany
j.schowalt09@uni-kl.de

Sylwia Kopczyńska
Poznań University of Technology
Poznań, Poland
sylwia.kopczynska@cs.put.poznan.pl

Svenja Polst
Fraunhofer IESE
Kaiserslautern, Germany
svenja.polst@iese.fraunhofer.de

Sadaf Alvani
University of Kaiserslautern
Kaiserslautern, Germany
alvanis@rhrk.uni-kl.de

Abstract

The growing interest of the requirements engineering (RE) community to elicit user requirements from large amounts of available online user feedback about software-intensive products resulted in identification of such data as a sensible source of user requirements. Some researchers proposed automation approaches for extracting the requirements from user reviews. Although there is a common assumption that manually analyzing large amounts of user reviews is challenging, no benchmarking has yet been performed that compares the manual and the automated approaches conderning their efficiency. We performed an expert-based manual analysis of 4,006 sentences from typical user feedback contents and formats and measured the amount of time required for each step. Then, we conducted an automated analysis of the same dataset to identify the degree to which automation makes the analysis more scalable. We found that a manual analysis indeed does not scale well and that an automated analysis is many times faster, and scales well to increasing numbers of user reviews.

1 Introduction

Developers of consumer products face the challenge of receiving increasing amounts of user feedback about their products. This feedback has been shown to contain useful experience-based information from users about their functional and non-functional requirements [GKH+17], so it can help improve the products to gain market advantage and retain user satisfaction [GSA+17]. A product developing company may choose to hire external consultants to do market research or to try and save costs by assigning a team member to assess the user feedback,

among others by reading user reviews. But as the amount of user feedback grows, it becomes more difficult to put in the effort required to assess all user reviews. Theoretically, this makes automating user feedback analysis a promising alternative.

A growing body of research in requirements engineering (RE) has dedicated to understanding and classifying user feedback using natural language processing (NLP) techniques. The proposed methods and tools are usually evaluated by comparing the results from NLP analysis to truth sets based on a manual analysis of a smaller subset of the user feedback (e.g., [IH13, PDG+15, WM17]). The categorization is often one-dimensional (e.g., "positive", "negative", and "feature request" [IH13]). These studies often report the time spent by each annotator on classifying the subset. That time is the basis to conclude that an analysis performed entirely manually is not sensible. However, we do not know of any research that has looked into determining the actual scalability of manual analysis.

To determine the effort involved in performing a manual analysis of user reviews, we conducted an expert-based analysis of online user reviews of compact cameras. Then, we compared the results with the effort needed for automated analysis. Our study considered cameras that is software-intensive systems in which the boundaries between software and hardware are blurring. The first reason is that the up-to-date research has primarily focused on apps [PM13]. Secondly, due to the rising popularity of smart systems. With the study we sought to answer the following research questions:

RQ1: How well does the manual classification of requirements in user feedback scale in terms of required effort?

RQ2: How does automating some parts of the user feedback analysis process influence its scalability?

Section 2 of this paper discusses related work. Sections 3, 4 and 5 present the method, results and discussion of our content analysis. Section 6 describes threats to validity, and Section 7 summarizes and suggests possible future work.

2 Related Work

RE can considerably benefit from NLP techniques. For example, quality defects such as inconsistencies can be identified in documented requirements. Additionally, specifications of very- and ultra-large-scale systems can be better managed [RSW08] by having an ability to find and categorize requirements for instance. In recent years, a growing body of research evaluates the use of NLP techniques for requirements elicitation. Most of them aim at deriving requirements from various types of user feedback [GSA+17]. Our study extends this area by providing the perspective of efficiency of requirements elicitation by extraction from the user feedback.

Earlier studies have analyzed larger samples of user feedback from an RE perspective by categorizing contents. Pagano and Maalej [PM13] performed statistical analyses on app store reviews, and manually analyzed 1,100 reviews, reporting that around 30% of all reviews could be categorized as "requirements", including sentences containing bug reports and shortcomings. Guzman and Maalej [GM14] analyzed 2,800 user reviews and found feature requests in 10%. Panichella et al [PDG+15] tagged a sample of 1,421 sentences from app store reviews, from which non-informative reviews were already omitted, and classified 97.82% as either "information seeking", "information giving", "feature request" or "problem discovery". Guzman, Alkadhi, and Seyff [GAS16] manually analyzed 1,000 tweets, and in requirements-relevant categories found 0.90% bug reports, 1.50% feature shortcomings, and 0.10% feature requests. Guzman, Ibrahim, and Glinz [GIG17] manually analyzed 1,350 tweets from the support accounts of three apps, of which 42% could be classified as improvement requests. Kurtanović and Maalej [KM17] coded 1,020 user reviews (6,268 sentences; 7,288 sentences including title) on the user rationale concepts "issues", "alternatives", "criteria", "decisions", and "justifications", and found that 48% of the sentences and 86% of the reviews contained one or more of these concepts. Williams and Mahmoud [WM17] manually classified 4,000 tweets, and found that 27% expressed "bug reports" and another 24% contained "user requirements". Our study differs from the related work by analyzing results for software-intensive products, and making a more fine-grained analysis by not only considering features and sentiments, but also quality aspects (cf. [GKH+17]), general sentences, and sentence topics.

3 Method

Our analysis, based on Neuendorf's content analysis method [Neu02], consisted of the following steps:

Step 1: Select and obtain user reviews. As online user reviews to apps have received much attention (e.g., [GW13, GM14, GIG17, JM17, PM13, SAH+15]), we chose to analyze user reviews to smart software-intensive systems, as we see an increasing trend towards these systems. We assumed a case in which a company would

Table 1: Overview of analyzed cameras and number of user reviews available. The number of sentences is composed of the number of sentences in the title and in the contents.

Camera	User Review Count	Sentence Count	Avg. Sentence/Review
Panasonic Lumix DMC-7270	10	35	3.50
Olympus Stylus TG-8609	163	1,010	6.20
Canon PowerShot SX610 HS	59	335	5.68
Canon PowerShot D30	149	1,043	7.00
Nikon Coolpix AW130	123	685	5.57
Nikon Coolpix S7000	82	532	6.49
Sony Cyber-shot DSC-HX60	38	366	9.63
Total or average	**624**	**4,006**	**6.42**

Table 2: Overview of tags and their tagging criteria.

Tag	Criterion
Positive	This sentence contains a positive remark about the product
Negative	This sentence contains a negative remark about the product
Requested	This sentence contains a plea or wish regarding the product
None	This sentence is not relevant
Feature	This sentence is about a functional aspect (i.e., a feature or product characteristic)
(Quality characteristics)	Seven out of eight product quality characteristics as defined in the ISO 25010 standard [ISO11]
General	This statsentence is neither about a feature (product characteristic) nor a quality (non-functional aspect)

like to gain insights into what users want or need from currently popular pocket-sized compact cameras in the price range around 200 euros that have innovative or (software-based) smart functions. We selected 12 presently popular compact cameras. Ten of them were available on Amazon, from where we crawled the user reviews. To obtain an objective overview of all requirements for that particular camera, we intended to analyze all user reviews of each camera rather than choosing a sample. To reduce skewing, we omitted three cameras with deviating review counts and lengths; one camera had only 17 reviews with an average length of 12.59 sentences (versus 7.12 for all cameras), and the reviews about two other cameras provided about 2,000 sentences per camera, which would cause half the dataset of nine cameras to consist of two cameras. As a result, our dataset consisted of the seven cameras presented in Table 1.

Step 2: Structure and import user reviews. Because a user review usually covers several aspects (e.g., both positive and negative aspects), we split the reviews into separate sentences using a sentence splitter. Based on the punctuation used, the sentence splitter could also split compound sentences and sentences that include a listing into separate sentences. We then stored the list of sentences in a CSV file, which we loaded into a customized web-based tagging tool. Here, annotators could assign tags representing specific categories to which a sentence belongs, such as "positive" or "negative". This tool allowed the annotators to tag independently of one another.

Step 3: Define tagging categories and a tagging schema. The exact tags that could be assigned in the tool were determined based on previous methods that determined the valence ("positive" and "negative") of a sentence, along with the option "feature request" [GM14, GW13, IH13, MN15]. Based on our findings [GKH+17] that sentences also contain information about product qualities (i.e., non-functional aspects) in addition to product features (i.e., functional aspects), we introduced a second tagging level (see Table 2). The first level enabled the annotator to assign "positive", "negative", "requested" or "none" as main tag. In this level, "feature request" was changed to "requested" to also enable combinations with quality requests and general requests. If an annotator picked an option other than "none", a sub-level tag was assigned. This sub-level contained the options "feature", seven of the eight software product characteristics from the ISO 25010 standard [ISO11] ("maintainability" was omitted because users do not have insight into the aspects of this development-specific quality [GKH+17]), and the category "general" for positive, negative, and requesting sentences that do not specifically concern a product characteristic or quality. The two levels together can provide combinations such as "feature requests" ($requested \times feature$), or "negative sentences about usability" ($negative \times usability$). For a common understanding of the meaning of the tags, a tagging schema was based on an earlier study's schema with examples and signal words, and the annotators were trained on the ISO 25010 characteristics.

Step 4: Perform tagging. As the content was expected to have ambiguities, each sentence was annotated separately by three of this paper's authors; one researcher and two computer science students. The annotators selected and read each sentence, and assigned the tag that they found most suitable. More than one tag and

sub-tag per sentence were allowed for each distinctive part of a sentence that could be clearly cut in several distinct parts (e.g., containing independent clauses separated by "but" or "and", or because the sentence splitter failed to split it due to missing punctuation). In order to obtain benchmarking results, the annotators were instructed to tag as precisely as possible but without excessive deliberation, and to log their time.

Step 5: Reconcile and process tagging results. We exported the tagging results to an Excel file and assigned each sentence a tag that scored the majority of votes. For the main tag, there was a majority agreement on 3,946 of the 4,006 sentences (98.50%). After omitting the sentences assigned"none", there was an agreement on the assigned sub-tag for 1,387 of the 1,844 remaining sentences (75.22%). Sentences with multiple tags by multiple annotators were split, because it indicated that different portions of a sentence should be assigned its own tag. Sentences on which there was disagreement were manually reconciled. To calculate inter-rater reliabilities, we used Krippendorff's alpha [HK07] because the sub-tags had deliberately missing data as the annotators did not assign sub-tags if the main tag was "none". We considered the result our ground truth.

An annotator then analyzed sentences to identify the "topic" (i.e., a central keyword) and any attributes. For example, "I like the size of the display" has "display" as topic, and "size" as a positive attribute. Not all sentences have a topic. Similar topics (e.g., "screen" and "display") were merged. This was also done entirely manually to measure and assess the amount of effort involved in classifying data without the use of automation.

Step 6: Analyze data. The data resulting from the manual tagging were analyzed by calculating sums and frequencies of the source data. A more thorough analysis of the content analysis to determine the quality of the results is beyond the scope of this paper and will be discussed in another paper. We additionally calculated and analyzed the time measurements of all steps to estimate the scalability of a manual approach.

Step 7: Estimate automation potential. In addition to the steps suggested by Neuendorf [Neu02], we identified which steps can potentially be automated. Furthermore, the amount of time that could be saved by automation compared to the manual analysis was calculated.

4 Results

4.1 Descriptive Statistics

The dataset consisted of 624 user reviews, providing a total of 4,006 sentences on seven compact cameras (see Table 1). Each product received on average 89.1 user review and each user review consisted on average of 6.2 sentences. All three annotators considered 1,563 sentences (39.02%) as irrelevant. After reconciliation, a further 774 sentences (19.32% of total) were considered irrelevant, and 27 sentences (e.g., compound sentences) were split into two, three or four parts. This resulted in a final set of 1,708 sentences that were considered as relevant from an RE perspective. 534 reviews (85.58% of all reviews) contained all sentences. We expected low inter-rater reliability due to the inherent ambiguity of statements in user reviews (cf. [PDG+15]). We found moderate agreement on the main tags ($\alpha = 0.58$), which was nearly substantial (values of 0.61 and up) and fair agreement on the sub-tags ($\alpha = 0.37$), but approaching moderate agreement (values of 0.41 and up).

4.2 Manual Labor (RQ1)

The annotators needed on average 12:46 hours to tag the 4,006 sentences (see Table 3). The time was not continuous, as they required ample breaks in between sessions. Moreover, the speed of annotation is variable, which is confirmed by the strong negative correlation between session length and average annotation speed (r = -0.88). It suggests that the performance decreases as vigilance wanes. More specifically, this explains the large difference in the performance between annotators 1 and 3.

Table 3: Overview of the performance of the various annotators.

Annotator	Total duration	Avg. session duration [h]	Avg. sentences/minute	Min;Max sentences / session
Annotator 1	08:37:00	00:16:41	7.92	45; 270
Annotator 2	13:18:00	00:36:16	4.84	43; 445
Annotator 3	16:23:00	01:10:13	4.08	110; 733
Average	**12:46:00**	**00:41:03**	**5.61**	**482.67**

We determined the effort per process step (presented in Table 4) to assess the scalability of manual analysis. For each step we identified *fixed* timespans (constant for the process), and *variable* time amounts that depends

Table 4: Composition of time per process step in a manual analysis. The values of step 1a and step 4a present the combined duration of the work performed by three persons, each spending on average a third of that time. Steps 2b until 5a are not considered "automatable" as they would be redundant in an automated procedure.

Step	Duration [h]	Automatable	Variable
1a. Select products	01:30	No	Products
1b. Gather hyperlinks	00:15	No	Products
1c. Obtain (crawl) user reviews	00:06	Fully	User reviews
2a. Apply sentence splitter	00:05	Fully	Sentences
2b. Export file	00:05	N/A	Sentences
2c. Develop, deploy& test tagging tool & environment	03:30	N/A	N/A
2d. Configure tagging tool	00:15	N/A	N/A
2e. Import user reviews	00:30	N/A	N/A
3. Define tagging categories & tagging schema	00:40	N/A	Sentences
4a. Perform tagging	38:18	N/A	Sentences
4b. Adjust export criteria & export results	02:00	N/A	Sentences
5a. Reconcile results	08:00	N/A	Sentences
5b. Process results	15:00	Fully	Sentences
6. Analyze data	18:00	Partially	Visualizations
Total [hours]	**88:14:00**		

on how much user feedback is analyzed. The variability can be expressed as a relationship with the number of products, reviews, or sentences analyzed.

To assess whether this manual work can be scaled to large amounts of user feedback, we had tracked the time needed to perform each step (see Table 4). For the sake of simplicity, we proposed to model the total time as a linear function of data. One of the consequences of this approach is that we considered the combined annotation time from Table 3 as a continuous timespan, disregarding the need for frequent breaks or the decrease in annotation performance over time.

In a joint discussion, we determined the share of fixed and variable time spans needed for completing each step based on our experience. For example, in step 1a, a group of three persons chose the products to be analyzed, amounting to on average 9 minutes per person, or 27 minutes in total. This was the fixed time, leaving 63 minutes of variable time to look up and decide upon the inclusion of seven products, which added a variable time of 9 minutes per product. Note that we have subtracted the additional time spent to obtain the ten products we had initially identified. This results in the formula $T_{step1a} = 27 + 9N_P$, where T is the time required in minutes, and N_P is the total number of products. We repeated this process for all steps. We then calculated the summed values for each step, which in the case of step 1 is the sum of steps 1a, 1b, and 1c. For the later steps, we also included the variables N_R for the number of user reviews and N_S for the number of sentences.

- $T_{step1} = \sum_{i=a}^{c} T_{step1,i} = 27 + 9N_P + 2.14286N_P + 0.00962N_R = 27 + 11.14286N_P + 0.00962N_R$
- $T_{step2} = \sum_{i=a}^{f} T_{step2,\ i} = 0.00125N_S + 2 + 0.00075N_S + 210 + 15 + 15 + 0.00374N_S = 242 + 0.00574N_S$
- $T_{step3} = 40$
- $T_{step4} = \sum_{i=a}^{b} T_{step4,i} = 0.57364N_S + 90 + 0.00749N_S = 90 + 0.58112N_S$
- $T_{step5} = \sum_{i=a}^{b} T_{step5,i} = 0.11982N_S + 0.22466N_S = 0.34448N_S$
- $T_{step6} = 1,000 + 0.01997N_S$

Together, they form the following formula:

$$T_{total(manual)} = \sum_{i=1}^{6} T_{step(i)} = 1,399 + 11.14286N_P + 0.00962N_R + 0.95132N_S$$

The number of products influences only the two short steps 1a and 1b; the influence of user reviews number is negligible. But the number of sentences affects the analysis duration. In addition, at least 1,399 minutes (23.32 hours or 2.9 8-hour person days) are needed for preparing and post-processing the manual analysis.

Our analysis used 7 products, 624 user reviews, and 4,006 sentences. Based on the proposed formula that is $T_{total(manual)} = 1399 + 11.14286 \times 7 + 0.00962 \times 624 + 0.95132 \times 4,006 = 1399 + 78.00 + 6.00 + 3810.99 = 5,294$ minutes (88:14 hours), reflecting the total in Table 4. It assumes three annotators perform the tagging, because with one annotator the results can be biased, and with two annotators it is difficult to resolve conflicting results.

Based on the formula, we can determine that each additional sentence adds on average 57.08 seconds to the

Table 5: Composition of time per process step in the automated analysis. The numbering of the step has been aligned with Table 2. Steps 1a and 1b are still performed manually, and step 6 is performed partially manually.

Step	Duration [h]	Automatable	Variable
1a. Select products	01:30	No	Products
1b. Gather hyperlinks	00:15	No	Products
1c. Obtain (crawl) user reviews	00:06	Fully	User reviews
2a. Apply sentence splitter	00:05	Fully	Sentences
X. Match language patterns	01:00	Fully	Sentences & Pattern Combination
5b. Process results	00:30	Fully	Sentences
6. Analyze data	18:00	Partially	Visualizations
Total [hours]	**21:26:00**		

analysis process. Of this time, 34.42 s (60.30%) are spent on tagging, 13.48 s (23.62%) on processing results, 7.19 s (12.60%) on reconciliation, and the remaining 1.99 s (3.49%) on data import, export, and analysis. This shows that the steps following tagging also still require a great deal of manual work.

To assess the actual scalability of manual work, we calculated the increase in time required for the manual analysis of sentences. To this end, we performed a correction for the increase in the number of products and reviews by assuming the average review lengths and the average number of reviews per product from our dataset. This shows that even a threefold number of sentences (12,000) will already require 217:78 hours of work (27.2 person days or 1.24 person months), or a workload of 41.3% FTE when evenly divided over three persons. This work is required for about 1,935 user reviews, which some apps receive on a daily basis.

4.3 Automation Benchmark (RQ2)

To make the comparison between the manual and automated approach, we compared the duration of the steps required to conduct each one. We considered all steps of the processes, regardless of whether they are automated or performed manually. We also assumed that all the steps are taken when a process is already in place. We had not measured the time for preparation, such as developing and configuring tooling, or defining the classification categories of the manual or automated approaches. The reason behind our assumption is that we could not reliably estimate the time those initial activities require as they greatly depend on the context.

When automating the analysis process, we could determine that steps 2b until 5b are replaced by two steps: sentence matching, and post-processing to find topics and attributes. The steps and their duration are shown in Table 5. The language pattern matching duration depends not only on the number of sentences matched but also on the number of possible combinations of patterns to be matched to the sentences. An analysis with 41 simpler language patterns could be performed at 160 sentences per second. With growing number of language patterns that also get increased in complexity, along with writing the results to the database, this speed is considerably reduced. We chose to over-approximate the duration of pattern matching over 4,006 sentences to last one hour (i.e., 1.11 sentences per second), which is way over the actual duration of the step. The processing of results with part-of-speech tagging is a quicker process and lasts approximately 30 minutes for this dataset.

Assuming the above estimations, the steps had the following durations:

- $T_{step1} = \sum_{i=a}^{c} T_{step1,i} = 27 + 9N_P + 2.14286N_P + 0.00962N_R = 27 + 11.14286N_P + 0.00962N_R$
- $T_{step2} = T_{step2,a} = 0.00125N_S$
- $T_{stepX} = 0.01498N_S$
- $T_{step5} = T_{step5,b} = 0.00749N_S$
- $T_{step6} = 1000 + 0.01997N_S$

Together, they form the following formula:

$$T_{total(automated)} = 1,027 + 11.14286N_P + 0.00962N_R + 0.26086N_S$$

We can see that the process for 4,006 sentences is reduced from 88:14 to 21:26 hours, and each additional sentence adds only 2.62 seconds to the total time. Of these 2.62 seconds, 1.19s (45.71%) is spent on the crawling step, 0.90s (34.29%) on the pattern matching, 0.45s (17.14%) on processing the results, and a negligible part on sentence splitting (0.07s; 2.86%).

Based on the formula for $T_{total(automated)}$, we can determine that analyzing a threefold number of sentences (12,000 sentences) increases the total time spent by only 8.61 hours to 30:03 hours, of which the largest portion (18 hours) is still spent on performing the largely manual interpretation stage of the analysis results.

5 Discussion

Our study showed that manual elicitation of requirements from user reviews is possible, but very time-intensive. For only 324 user reviews, classifying their 4,006 sentences took on average 12:46 hours of non-continuous work. This is in line with the reported effort in related works, in which an annotator spent 13:36 hours to classify topics of 327 reviews [GW13], between 8:00 and 12:30 hours per annotator to categorize 900 reviews [GM14], or an average of 13.6 hours to label 1,350 tweets [GIG17]. A manual analysis of even larger amounts will, therefore, be very tedious [FLL+13]. An essential finding of this study is that especially the number of sentences determines the effort of the analysis. We observed that each additional sentence is worth about a minute of the analysis manually performed by three persons.

We additionally found tagging time to strongly correlate with session length, suggesting that a human annotator gets exhausted after some time. In addition to this, being a logical consequence of concentrated reading, user reviews have many redundancies. Such work becomes tedious and uninteresting for a human annotator, which would likely lead to less precise work while performing the categorization and processing results. For optimal performance, tagging sessions shall be short with ample breaks, limiting the tagging activity to about two hours (about 20% FTE or 600 sentences) on a given work day. On a monthly basis, this would translate to roughly 13,200 sentences or about 2,057 reviews per month (assuming no learning effect) that can be three persons would be able to handle efficiently. Moreover, the other process steps would require additional work time. The manual analysis of reviews to an app such as the Amazon Video app, with on average 2,280 user reviews per month, would according to our calculation require 1.48 person months of intensive work. From this perspective, this number of user reviews seems too large to analyze sensibly manually.

The relationship between the amount of user feedback to analyze and the total time of the analysis that can be expressed as the polynomial might seem that manual analysis do scale well. However, from the business perspective, it must be taken into account that human work is not cheap. Our first argument is that the coefficients that describe the relationship have high values (one sentence is worth around a minute of work). Secondly, the analysis is a repetitive and cognitively demanding task and the associated need for taking breaks strongly affects the practical scalability. Thus, the upper boundary for manual analyses to elicit requirements, according to our calculations, seems to be around 2,000 user reviews per month. For greater amounts, the potential to continue performing the analysis manually should seriously be called into question (**RQ1**).

Table 5 shows that when introducing automation into the analysis of user feedback, the steps to crawl, process, and annotate sentences become obsolete, and especially the categorization of sentences gets drastically shortened (**RQ2**). The product selection remains a manual task. The presentation and interpretation of the results are tool-supported activities that, currently, also involve quite much manual work.

Reliable measurements of the time required to implement tooling, it would be possible to determine a break-even point for the investment in tooling. The investment is likely to pay itself off very quickly as the number of analyzed sentences increases, even if the automation is still being fine-tuned in the process. The repetitive nature of the work makes it an unpleasant task for humans, but filtering out irrelevant and repetitive contents would help. However, machine learning will have some difficulty with finer nuances in language, including slang, expressions (e.g., “I got a lemon”) and sarcasm.

6 Threats to Validity

We discuss the validity threats to our study based on the guidelines by Wohlin et al. [WRH+12].

Construct validity. The opinions of users are inherently ambiguous, among others because natural language is informal, and authors of user reviews can be computer laypeople, non-native English speakers or just write without much attention. This made a tagging difficult and explains the considerable amount of time dedicated it took. Another threat is that the annotators had experience with compact cameras, but not as users of the analyzed models, which might have caused misinterpretations, and occasionally required re-reading sentences for better understanding. To reduce bias and misinterpretation, we chose annotators who speak English fluently, who received a training on the ISO 25010 quality characteristics, iteratively refined the tagging schema, and performed a pilot tagging. We have not made a comparison of the quality of the outcomes, although it can be assumed that especially the recall of the automated analysis will be less than of three human raters.

Internal validity. We selected the reviews of the domain and type of product arbitrarily. The goal behind this decision was to investigate a set of opinions regarding particular software-intensive systems. To mitigate the subjectivity of the experts in assigning tags, three reviewers performed the tagging. We omitted products from our sample that were not on Amazon or had extreme deviations from the mean to reduce the difference in

numbers and lengths of reviews per camera. For each of the remaining cameras, we analyzed all of the reviews available at the time of crawling.

External validity. The user reviews were about software-intensive products, and were taken exclusively from Amazon. We argue that this data source contains representative opinions of the cameras. Although the results may not be directly comparable to the tagging of user reviews to mobile apps, we did see commonalities in the time required to perform the tagging. However, reviews for these kinds of products may be longer, which possibly has caused our formula to somewhat underestimate the influence of the number of reviews on the analysis time. Next, the effort of our three annotators shall be treated like the upper band, as experienced annotators might work faster. However, the times we reported were comparable to that in other scientific works. We have assessed the scalability of automation on only one tool, but have been generous in estimating the time it requires to perform an analysis, so that we are confident that we have been able to cover the performance of many comparable algorithms and tools.

Conclusion validity. A potential threat relates to the size of our sample of data. The set contained opinions of seven cameras with 624 reviews. We have simplified the formula by assuming it to be linear, although in practice, several aspects are not necessarily linear (e.g., human annotators not being able to perform their work continuously).

7 Conclusions & Future Work

In this paper, based on the manual analysis of online user reviews of smart compact cameras, we determined that it is possible to perform the process of deriving requirements from user feedback manually. However, due to the required effort and fatigue occurring, we found that manual analysis does not scale well. Additionally, manually analyzing 2,000 user reviews per month seems to be the upper limit when continuously monitoring the user feedback a product receives (e.g., to identify problems and ideas for enhancements).

The **main contribution** of this paper is twofold. Research towards NLP has often expressed the assumption that manually analyzing large amounts of user feedback is hopeless work, but this claim was based on gut feeling. Our study is the first investigation we know on the effort involved in performing such analysis. We have provided tentative evidence in favor of perception. The second contribution is that we have put an approximate number on the limits of manual analysis, along with a formula that distinguishes between the fixed and variable effort. It might support a company in assessing the effort needed to analyze user feedback of their product(s) and in deciding whether humans shall perform the work.

Hence, we have shown that NLP techniques and algorithms are indispensable for being able to monitor what users write about a product. By performing a benchmarking, we have shown that even with very cautious estimates, these techniques scale well to large amounts of user feedback. When introducing automation, manual steps regarding data import/export, tagging and categorization can be omitted or receive an efficiency boost. However, some manual steps remain, and currently, much effort is still needed to develop and fine-tune tool-support. Although automated approaches are often criticized for missing aspects that are hard to classify without a better understanding of the context and the finesses of a language (i.e., resulting in a lower recall), human annotators can be unintentionally biased when reading sentences, possibly emphasizing less important topics. It then remains the question if the speed and convenience of using NLP techniques outweighs having a loss of recall, which is currently often found to be around 0.70 (e.g., [WM17])

One of the direction for future research could be to propose more accurate formulas modelling the relationship between the total effort and the amount of user feedback. However, we would rather suggest that future research focuses on improving the results from the automated analysis, and on better deriving user requirements from the results. At best, a more thorough analysis could suggest a formula to determine the break-even point between automation and manual analysis, which would help practitioners determine whether they should use automation to elicit requirements from user feedback.

Acknowledgment

The research described in this paper was performed in the project Opti4Apps (grant no. 02K14A182) of the German Federal Ministry of Education and Research.

References

[FLL+13] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, N. Sadeh. Why People Hate Your App: Making Sense of User Feedback in a Mobile App Store. *19th ACM SIGKDD Intl Conference on Knowledge Discovery and Data Mining*, pp. 1276–1284. ACM, 2013.

[GW13] L. V. Galvis Carreño, K. Winbladh. Analysis of User Comments: An Approach for Software Requirements Evolution. *35th Intl Conference on Software Engineering*, pp. 582–591. IEEE Press, 2013.

[GKH+17] E. C. Groen, S. Kopczyńska, M. P. Hauer, T. D. Krafft, J. Doerr. Users–The Hidden Software Product Quality Experts? *25th Intl RE Conference*, pp. 72–81. IEEE, 2017.

[GSA+17] E. C. Groen, N. Seyff, R. Ali, F. Dalpiaz, J. Doerr, E. Guzman, et al. The Crowd in Requirements Engineering: The Landscape and Challenges. *IEEE Software, 34*(2), pp. 44-52. IEEE, 2017.

[GM14] E. Guzman, W. Maalej. How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews. *22nd Intl RE Conference*, pp. 153–162. IEEE, 2014.

[GAS16] E. Guzman, R. Alkadhi, N. Seyff. A Needle in a Haystack: What Do Twitter Users Say about Software? *24th Intl RE Conference*, pp. 96–105. IEEE, 2016.

[GIG17] E. Guzman, M. Ibrahim, M. Glinz. A Little Bird Told Me: Mining Tweets for Requirements and Software Evolution. *25th Intl RE Conference*, pp. 11–20. IEEE , 2017.

[HK07] A. F. Hayes, K. Krippendorff, Answering the Call for a Standard Reliability Measure for Coding Data. *Communication Methods and Measures, 1*, pp. 77–89, 2007.

[IH13] C. Iacob, R. Harrison. Retrieving and Analyzing Mobile Apps Feature Requests from Online Reviews. *10th International Conference on Mining Software Repositories*, 41–44, 2013.

[ISO11] International Organization for Standardization. *ISO/IEC 25010:2011 – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models.* ISO, 2011.

[JM17] N. Jha, A. Mahmoud. Mining User Requirements from Application Store Reviews Using Frame Semantics. In: P. Grünbacher, A. Perini (eds.), *Requirements Engineering: Foundation for Software Quality, LNCS, vol. 10153*, pp. 273–287. Springer, 2017.

[KM17] Z. Kurtanović, W. Maalej. Mining User Rationale from Software Reviews. *25rd Intl RE Conference*, pp. 53–62. IEEE, 2017.

[MN15] W. Maalej, H. Nabil. Bug Report, Feature Request, or Simply Praise? On Automatically Classifying App Reviews. *23rd Intl RE Conference*, pp. 116–125. IEEE, 2015.

[Neu02] K. Neuendorf. *The Content Analysis Guidebook.* Sage Publications, 2002.

[PM13] D. Pagano, W. Maalej. User Feedback in the AppStore: An Empirical Study. *21st Intl RE Conference*, pp. 125–134. IEEE , 2013.

[PDG+15] S. Panichella, A. Di Sorbo, E. Guzman, C. Visaggio, G. Canfora, H. Gall. How Can I Improve My App? Classifying User Reviews for Software Maintenance and Evolution. *31st Intl Conference on Software Maintenance and Evolution*, pp. 281–290, 2015.

[RSW08] B. Regnell, R. B. Svensson, K. Wnuk. Can we Beat the Complexity of Very Large-Scale Requirements Engineering? In: B. Paech, C. Rolland (eds.), *Requirements Engineering: Foundation for Software Quality, LNCS, vol. 5025*, pp. 123–128. Springer, 2008.

[SAH+15] F. Sarro, A. A. Al-Subaihin, M. Harman, Y. Jia, W. Martin, Y. Zhang. Feature Lifecycles as They Spread, Migrate, Remain, and Die in App Stores. *23rd Intl RE Conference*, pp. 76–85. IEEE, 2015.

[WM17] G. Williams, A. Mahmoud, Mining Twitter Feeds for Software User Requirements, *25th Intl RE Conference*, pp. 1–10. IEEE, 2017.

[WRH+12] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén, *Experimentation in Software Engineering.* Springer, 2012.