On the Potentials of Realtime Sentiment Analysis on Text-Based Communication in Software Projects

Lennart Schroth[1], Martin Obaidi[2], Alexander Specht[2], and Jil Klünder[2(✉)]

[1] Leibniz Universität Hannover, Hanover, Germany
info@lennart-schroth.de
[2] Software Engineering Group, Leibniz Universität Hannover, Hanover, Germany
{martin.obaidi,alexander.specht,jil.kluender}@inf.uni-hannover.de

Abstract. Sentiment analysis is an established possibility to gain an overview of the team mood in software projects. A software analyzes text-based communication with regards to the used wording, i.e., whether a statement is likely to be perceived positive, negative, or neutral by the receiver of said message.

However, despite several years of research on sentiment analysis in software engineering, the tools still have several weaknesses including misclassifications, the impossibility to detect negotiations, irony, or sarcasm. Another huge issue is the retrospective analysis of the communication: The team receives the results of the analysis at best at the end of the day, but not in realtime. This way, it is impossible to react and to improve the communication by adjusting a message before sending it.

To reduce this issue, in this paper, we present a concept for realtime sentiment analysis in software projects and evaluate it in a user study with twelve practitioners. We were in particular interested in how realtime sentiment analysis can be integrated in the developers' daily lives and whether it appears to be helpful. Despite the still missing long-term case study in practice, the results of our study point to the usefulness of such kind of analysis.

Keywords: Sentiment analysis · Social aspects · Software project · Team mood · Realtime feedback

1 Introduction

In modern software development, several project leader strive to obtain an overview of what is going on in the teams [8,17]. This overview is not only limited to the technical aspects such as development progress, but also includes social aspects as they have been proven to influence the productivity of the team [4–6,24]. So-called sentiment analysis tools are one possibility to observe text-based communication that are widely used in software engineering [28]: Applying

Published by Springer Nature Switzerland AG 2022
R. Bernhaupt et al. (Eds.): HCSE 2022, LNCS 13482, pp. 90–109, 2022.
https://doi.org/10.1007/978-3-031-14785-2_6

such tools to chats or other communication channels returns the polarity of each statement, i.e., whether it is positive, negative, or neutral, or an aggregated value for the whole team (e.g., [3,10,11]). Assuming that the used language reflects the general mood of a team member (happy team members are likely to communicate more friendly and positive, whereas a dissatisfied team member is likely to write more negatively), these values allow project leaders or managers to get an overview of the team mood without investing much time [3]. Due to the involvement of tools, this mood is kind of objectively measured.

However, such sentiment analysis tools have some weaknesses ranging from general problems with the accuracy of the detected polarity [12,21] to the impossibility to handle negotiations, irony, sarcasm, and the like [2,9,13,14,25,26]. Another problem is the time frame: The text-based communication is analyzed afterwards, e.g., at the end of a day, to draw conclusions about what was going on during the day (e.g., [7,23]). Hence, realtime interventions, i.e., adjusting the negative text-based message to be less negative, are almost impossible. However, another problem in software projects is the missing awareness of team members about the possible influences of their (interpersonal) behavior, including inadequate communication [17]. Even more, they are often not aware about how they communicate, i.e., whether it is likely that other team members perceive their text-based communication as positive and friendly or negative and unfriendly [16,17]. Consequently, providing them with information about how their message might be perceived before they send it to the rest of the team allows them to adjust it. Such kind of realtime feedback allowing team members to react has proven to be valuable for software development team [19].

As a first attempt to solve this problem, in this paper, we introduce a concept for realtime sentiment analysis for text-based communication between developers. This concept enables an analysis of all kind of input via the keyboard and returns the sentiment polarity of the input (independent of whether it is a word or a statement) so that the sender of the message receives a sentiment score representing how the receiver might perceive his text. This allows the sender to adjust the used language if necessary (e.g., if the message appears to be unintentionally negative). We evaluate the concept according to its applicability and its possible usefulness in a user study with twelve practitioners. Summarizing, we contribute the following key findings:

- The study participants agree that the core idea of realtime sentiment analysis is useful for industry.
- They would be willing to share their personal results in order to calculate an average team mood despite some security concerns.
- The participants' suggestions to improve the software point to an integration of established techniques that are already used in retrospective sentiment analysis into the concept of realtime sentiment analysis.
- We as experimenters observed an increasing awareness on how the participants communicate over the course of the study.

Context. This paper is based on the master's thesis [29] by Lennart Schroth with the title *"Concept for a preventive sentiment analysis of messages in software development teams"*.

Outline. This rest of this paper is structured as follows: In Sect. 2, we present background and related work. The concept of realtime sentiment analysis is outlined in Sect. 3. Section 4 summarizes our study design with the research questions, the instrument development, the data collection, and the data analysis. In Sect. 5, we present our results which we discuss in Sect. 6. Section 7 concludes the paper.

2 Background and Related Work

The idea of realtime sentiment analysis extends the core idea of sentiment analysis that is an established and frequently used method in software engineering [20,28]. There are several existing tools to analyze text-based communication, some of which are especially designed for the use in the software engineering domain [1,3,10,11,14,15].

The application of sentiment analysis in software engineering is not new [20,28]. Already in 2010, Thelwall et al. [30,31] presented SentiStrength which is a lexicon-based tool that searches for positive or negative words in a sentence (according to a pre-defined lexicon) and calculates the overall polarity based on these words. SentiStrength-SE is an adjusted version of SentiStrength that is specifically designed for the software engineering domain [14]. Comparing SentiStrength and SentiStrength-SE, Islam et al. [14] showed that the latter one provides more accurate results in software engineering contexts.

Klünder et al. [16] developed a sentiment analysis tool for classifying textual communication in a development team. They collected their data based on digital correspondence in German between developers working in the industry [16]. Their tool can detect polarities and is based on different machine learning algorithms like support-vector machine, random forests and an evolutionary algorithm for selecting suitable metrics to achieve the best performance.

Herrmann and Klünder [10,11] developed a sentiment analysis tool called SEnti-Analyzer which can analyze text-based communication data for their polarities as well as audio recordings and meetings in real time. The tool also assigns polarities and works with English and German language. SEnti-Analyzer includes several lexicon-based as well as machine learning based tools and combines them in a majority voting, which consists of the median label of each tool.

Calefato et al. [3] developed a tool called Senti4SD and enabled training and also classification of models specific to the SE domain. They implemented both a specialized word lexicon and also a support-vector machine. They were able to classify an input document in one of the three polarities *positive, negative, and neutral*.

Novielli et al. [26] also compared different sentiment analysis tools in a software engineering context. They tested the tools on data from GitHub, Jira, and

Stack Overflow. According to their results, lexicon-based methods provide better results, with, e.g., SentiStrength-SE having an accuracy of 75 to 80%.

Zhang et al. [33] compared the performance of different pre-trained neural network models (e.g., RoBERTa [22]) with those tools using more classical machine learning approaches or lexicon based tools (e.g., SentiStrength-SE [12]). They evaluate the classification accuracy of all these tools on many the data sets. They observed that the RoBERTa model [22] most often had the highest scores on average among the pre-trained transformer models.

In their replication study, Novielli et al. [27] explained some sentiment analysis tools (e.g. Senti4SD [3]) in great detail and described the underlying data. They also calculated an interrater agreement between sentiment analysis tools with each other and also with manual annotations from a gold standard of 600 documents. Based on their results, they suggest platform-specific tuning or retraining for sentiment analysis tools [27].

However, all these tools or studies did not perform a realtime sentiment analysis. But in order to be able to address the social factors such as negative mood in teams at an early stage, realtime sentiment analysis could be a solution. Our approach is based on such existing tools as they have been proven to provide adequate results with a sufficient accuracy.

3 Concept: Realtime Sentiment Analysis in Software Projects

Our concept for realtime sentiment analysis strives to detect the mood transported in a text-based message. So far, sentiment analysis on text-based communication requires time afterwards and the results are presented retrospectively [28]. The concept includes a window that is displayed in parallel to the already used communication software and calculates and visualizes a sentiment score based on the typed words. With our approach, we want to minimize the time required to analyze the message by presenting the results before sending a message, locally on a terminal without central analysis service. This helps highlighting how a receiver of a message may perceive it (and allows adjusting the wording if necessary). Resulting from this line of thoughts, our approach consists of the following three steps:

1. **Data collection:** We need to collect the written messages either in one specific channel (e.g., Microsoft Teams) or in general.
2. **Data analysis:** The collected data needs to be analyzed using different (already existing) mechanisms.
3. **Data processing:** We need to visualize and evaluate the results of the analysis.

These three steps result in the following requirements for the software system that implements our concept:

– The software tool shall run locally on a terminal device (offline).
– After having started the tool, it analyzes all messaging inputs from the user.
– The tool shall provide the results of the analysis in a short time (<1 s).
– All user data has to be deleted when stopping the tool.

These steps and the requirements should be implemented as individual modules. A high degree of modularity and interchangeability should be ensured in order to enable future improvements with low effort. For further improvements, we wanted to allow the deactivation, editing, or replacement of modules while the system is running. For this reason, an Internet of Thinks (IoT) protocol was used to ensure future-oriented data traffic. Message Queuing Telemetry Transport (MQTT) is a solution for short messages between different services, supervised by the MQTT-Broker as node for data traffic and quality of service[1]. The dependency and connections of the individual modules is shown in Fig. 1. In the following, we describe each of the steps in more detail.

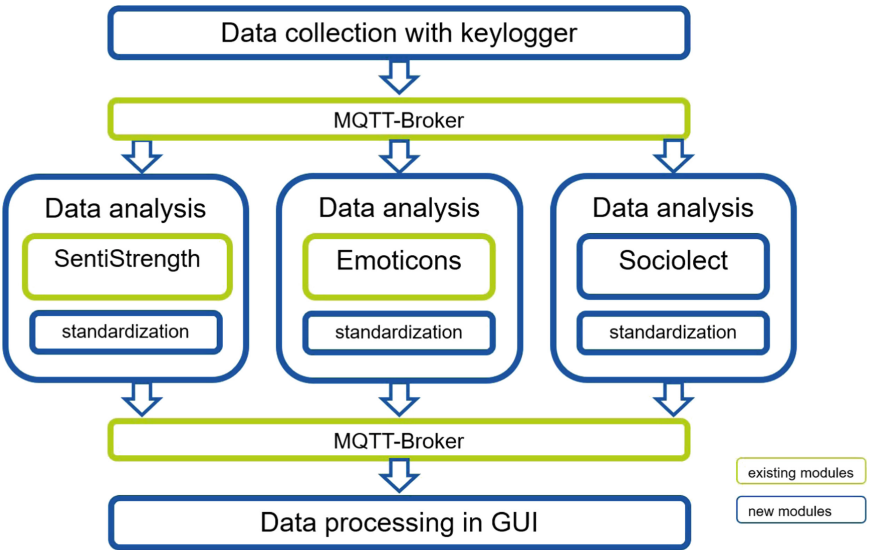

Fig. 1. Concept overview realtime sentiment analysis

3.1 Data Collection

In order to analyze different kind of text-based communication, we need to capture text or drafts of a message in realtime. As communication varies remarkably across teams (while one team prefers the communication via mail, while another

[1] https://mqtt.org/.

one prefers group chats [18]) and due to the wide variety of communication tools used in teams, we decide to use a keylogger that collects all types of keyboard inputs in the background. The keylogger helps identifying and collecting every input made using the keyboard and processes the input to be usable for the next step. The collected characters from the keyboard input are transformed to sentences to analyze the message as a whole instead of word-by-word. These were published on a MQTT-Broadcast channel respectively MQTT-Topic.

3.2 Data Analysis

For the data analysis, we use messages as input. For this step, we use a combination of existing tools consisting of an established sentiment analysis tool, the analysis of emoticons, and the analysis of the sociolect. Each of these methods was realized as a module to allow to activate or deactivate single tools. The following tools were considered for the sentiment analysis.

1. **SentiStrength**
   SentiStrength is a tool that detects mood in sentences by inferring positive or negative utterances from the polarity of words. E.g. "Good (+1) work". This tool has already been used in various studies and classifies quite reliably. Due to the evaluation of our approach in German companies with German as main language, we integrated the German version of SentiStrength in our concept. We integrate two versions of SentiStrength: SentiStrengh-DE (for German language) and SentiStrength-EN (for English language).
2. **Emoticons**
   An emoticon is the combination of different characters used in short messages to express the mood. E.g. "(^ ^)" (+1) ; "=(" (−1) ; "o.O" (0).
   We collected those emoticons in a look-up table with a sentiment value. The implemented script analyzes the incoming messages and calculate an average if multiple emoticons are used.
3. **Sociolect**
   During the pre-study, some statements were not recognized by SentiStrength, since they were neutral in terms of the polarities of the words, but are perceived negatively or positively in teams. A keyword-based analysis tool was then developed to classify such statements. In order to adapt this concept to different user groups, individual statements can be added manually to a lookup table.

The individual analyses had to be standardized for further data processing. In this context, the sentiment values were converted to a range from −1 (negative) to +1 (positive). In order to recognize why a message was interpreted in what way, the analyses also presents a reason for a calculated value.

3.3 Data Processing

We developed a user interface (in C#) that can be opened alongside an existing messenger service, such as Microsoft Teams. This slim display collected and

displayed the results of the realtime sentiment analyses. In addition to displaying the individual analyses with explanation, an overall average was also calculated and shown central.

3.4 Prototype

We implemented these core ideas regarding the data collection, analysis, and processing in a software prototype. A screenshot of the general functionality is visualized in Fig. 2. In the upper half of the screen, we see the results of the different sentiment analyses over time. Below, we see the most recent sentiment score of the message, and an overview of explanations about how the score was calculated in the bottom. Note that, as we conducted the study in Germany, we also integrated the German version of SentiStrength and just translated the input for this screenshot. Typically, SentiStrength-DE would not be able to provide any information for English inputs. However, both analyses run in parallel. That is, if the used words are present in the German lexica, the German version is used. However, as computer science is a field in which English words are omnipresent, we also look into English lexica (provided by SentiStrength-EN) to increase the richness of the information. Nevertheless, this parallelity of the both languages introduces some difficulties as there are words that have different meanings in the languages. For example, the word "war" (past tense form of "be" in German) has a neutral meaning in German, but a negative meaning in English. Further research is required to solve such issues.

4 Study Design

In the following, we present our study design with the research questions and goals, the experiment structure, and the data collection.

4.1 Research Goal and Research Questions

Our overall research goal is to *evaluate realtime sentiment analysis with regard to its applicability and usefulness in practice*. In particular, we strive to answer the following research questions:

Research Question 1

> *How can realtime sentiment analysis be integrated in the daily lives of software developers?*

This question strives to analyze whether the concept presented before is useful for realtime sentiment analysis.

Research Question 2

> *How useful is realtime sentiment analysis perceived by practitioners?*

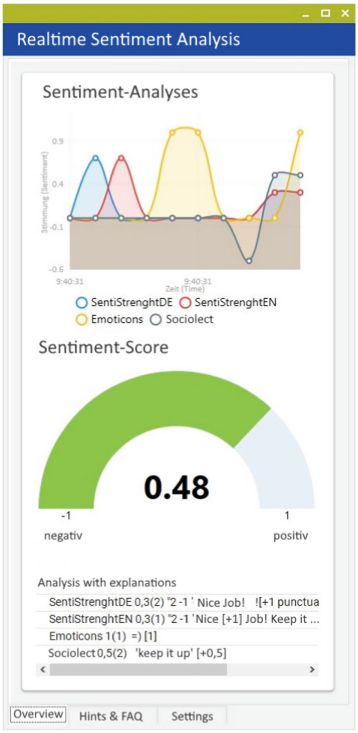

Fig. 2. Screenshot of the software prototype translated to English

This question clarifies if realtime sentiment analysis adds perceived value to software projects. In addition, it also shows how this added value looks like and how it can be achieved.

Research Question 3

> *Would team members be willing to share their aggregated data to allow an analysis on team level?*

This questions deals with the further data processing to not only reflect the mood of single team members, but also of the whole team, which is the main use case of retrospective sentiment analysis, but which needs to be adjusted to a realtime analysis.

4.2 Instrument Development

In order to evaluate the usefulness of realtime sentiment analysis, we implemented the aforementioned concept in a prototypical tool consisting of a *keylogger* to collect all keyboard inputs (in order to analyze them independently from the used messenger), three different analysis tools described in Sect. 3 and the visualization of all collected data over time.

4.3 Experiment Structure

We used the prototype to evaluate our concept of realtime sentiment analysis in an experiment with practitioners. The experiment consisted of five steps:

1. **Introduction (5 min):** We presented our overall idea of the realtime sentiment analysis and how the prototype works that they are about to use. We paid particular attention to a (brief) overview of the data processing and the further storage of the data.
2. **Questionnaire 1 (5 min):** The participants answered a questionnaire asking about demographic information (years of experience, their role in the team, etc.) and about their typical communication behavior (the use of different channels).
3. **Exploring the prototype (5 min):** The participants were asked to get to know the prototype in an exploratory fashion. In particular, they were asked to write something to get familiar with the main functionality of the software.
4. **Testing the prototype (10–20 min):** As the study strived to analyze the concept (i.e., the core idea of realtime sentiment analysis) rather than the prototype itself, we presented fictive scenarios that are possible use cases of the tool. As part of these scenarios, the participants were asked to write an answer to a received message. They were asked to write this answer in a positive, negative, or neutral way and to compare their perception with the response of the tool.
5. **Questionnaire 2 (5 min):** In order to collect the perceptions and opinions of the participants, they were asked to answer a second questionnaire to report on general feedback and their perceived usefulness of the tool. In addition, we integrated the "I wish, I like, I wonder" methodology to collect ideas for future research (*I wish*), core ideas that are good (*I like*), and potential for improvement or the need for explanations (*I wonder*).

4.4 Data Collection

As a first step, we collected data in a pre-study evaluating the overall concept and to allow for necessary improvements of the software prototype before testing it in a broader scope. In December 2021 four experts (project leader, quality assurance, or researchers with several years of experience with working in industry) participated in the pre-study. We used the feedback to make improvements to the prototype for analysis accuracy and user acceptance.

The pre-study followed the same process as described above, but strived to lay a foundation for the main study rather than to evaluate the idea from several viewpoints. That is, we conducted the pre-study to get a basis for the main experiment. We applied the think-aloud approach in order to collect as much feedback as possible. That is, participants were asked to verbalize their thoughts as fine-grained as possible. Resulting from the insights we gained during the pre-study, we improved the user interface and the sentiment analysis itself, as almost all participants faced problems with a few unclear classifications of messages: Team-specific statements were identified as neutral, although these were

perceived negative by the experts. As a consequence, we included an analysis of the sociolect[2] with the option to add typical phrases (with a polarity value) to a so-called look-up table to adjust the tool to the team-specific language. For example "keep it up" was analyzed as *neutral*, although it was meant to be *positive*, and "if it has to be" ("wenn es sein muss" in German) has no negative polarity words but is meant negatively.

The main study was then conducted in the end of January 2022 with twelve participants from industry that did not participate in the pre-study. Depending on the preferences of the respective participant, the study was either conducted in person or in an online setting. The experiments had a duration of 45–55 min. The twelve participants work in three medium-sized companies (with 50 to 250 employees) in the domain of information systems and telecommunication. All participants work in software development teams. Seven participants work as software developer, two as project leaders, two as technical designer, and one as an IT consultant. In total, four female and eight male team members participated in our study. On average, the participants had 10 years of experience with working in the IT domain with a minimum of one year and a maximum of 24 years.

During the study, we applied the think-aloud approach and took notes. In addition, the participants answered two questionnaires that were also used for the later data analysis.

These questionnaires consisted of three question on demographics, 13 items that were rated on a Likert scale, two open questions on communication behavior in general, and some space for feedback divided into *I wish* (ideas for improvement), *I like* (good core ideas that are useful), and *I wonder* (needs for explanations or adjustments).

4.5 Data Preprocessing and Data Analysis

To answer our research questions, we mainly used two data sources: (1) The answers to open questions on communication behavior and feedback in the questionnaire (together with our notes resulting from the think-aloud process) and (2) Ratings on Likert scales on statements to different aspects concerning our research questions.

(1) The answers to the open questions were coded using open coding. That is, we categorized all answers (both from the questionnaires and from our notes) as long as we were sure that each statement was assigned the best matching category. These categories then helped us draw conclusions on the usefulness, missing functionalities, and potential for improvement (among others). These open questions also include the feedback presented on the questionnaires.

[2] Sociolect is a form of language (non-standard dialect, restricted register) or a set of lexical items used by a socioeconomic class, a profession, an age group or other social group.

(2) For the Likert scales, we calculated the median values for each of the six statements presented in Table 1. Note that, although we present the English translation of the statements, the statements were provided in the native language of the participants.

Table 1. Items rated by the participants

ID	Item	Questionnaire
1	I receive sufficient information to solve my tasks	Q1
2	There are several team meetings for exchanging relevant information for the projects	Q1
3	I try to spread a positive mood in the team	Q1
4	I try to formulate messages in a positive way	Q1
5	The results of the analysis coincide with my perception	Q2
6	The different analysis mechanisms detected different moods adequately	Q2
7	Such kind of software might be helpful in the future	Q2
8	I had few security concerns when using the software	Q2
9	To improve the team mood, I would share the average results of the analysis of my data with the project lead	Q2
10	I would like to know the average team mood	Q2

5 Results

We conducted the study as described in the previous section. In the following, we present the results.

5.1 Characterizing the Participants

Regarding the typical information exchange in the companies, the participants reported on generally receiving sufficient information for their tasks (Statement: *I receive sufficient information to solve my tasks*; median: 4, min: 3, max: 5). That is, typically, they receive the required information to solve their tasks. In addition, there are several meetings to ensure sufficient information exchange (statement: *There are several team meetings for exchanging relevant information*

for the projects, median: 5, min 4, max: 5). The participants also try to spread a positive mood in the team (statement: *I try to spread a positive mood in the team*, median: 4, min: 3, max: 5) and to write positively (statement: *I try to formulate messages in a positive way*, median: 4, min: 3, max: 5).

At the time of the study (and likely influenced by the CoVID19-pandemic) participants report on using online-video meetings most often for communication (avg: 34,6%, SD[3]: 11,6%), followed by chat messages (22,1%, SD: 8,53%), e-mails (19,6%, SD: 11,4%), and face-2-face communication (18,3%, SD: 17,1%). Phone calls are barely used (5,4%, SD: 8,28%).

Text-based communication, that is analyzed by our tool, is used on average for 42% of the communication. Some participants use messages up to 60% to exchange information.

5.2 Perceived Accuracy of the Tool

Regarding the tools' accuracy, the participants were mostly satisfied (statement: *The results of the analysis coincide with my perception*, median: 4.5, min: 3, max: 5). However, note that this is only the perceived accuracy as we did not calculate some objective measure such as an interrater reliability between each participant and the tool. When analyzing social aspects in development teams, the perceived accuracy is much more important than the objective accuracy: If the participants' perception deviates too far from the results, they will likely neglect the results. Therefore, we opted for the perceived accuracy rather than for the objective measure. Nevertheless, we ensured objective accuracy by using established sentiment analysis tools.

In addition, the participants reported that the different mechanisms (SentiStrength, emoticons, and sociolect) detected the mood adequately (statement: *The different analysis mechanisms detected different moods adequately*, median: 4, min: 3, max: 5).

5.3 Perceived Helpfulness

The participants reported on a potential helpfulness of the software in future (statement: *Such kind of software might be helpful in the future*, median: 4.5, min: 3, max: 5). In order to improve the usability, they suggested to analyze the chats on project-level or to just consider specific chats of groups or teams, which is the current state of sentiment analysis, but retrospectively [28]. In addition, two participants recommended to provide the results comparable to a spell check underlining the words in red or green, respectively. Half of the participants were also interested in a live presentation of the results aggregated on chat-level right in the respective chat window.

[3] SD is the abbreviation for standard deviation.

5.4 Data Aggregation on Team Level

However, the participants had some security concerns when using the software (statement: *I had few security concerns when using the software*, median: 3, min: 2, max: 5). This also goes along with one participant reporting on the feeling of being monitored or observed. This feeling is also omnipresent if the aggregated results are shared with the project leader. Nevertheless, the participants reported on agreeing to share their data (statement: *To improve the team mood, I would share the average results of the analysis of my data with the project lead*, median: 4, min: 3, max: 5), as they are also interested in the average team mood (statement: *I would like to know the average team mood* , median: 5, min: 4, max: 5). However, ten out of twelve participants stated that they are only willing to share their average data if the whole team can see the results on team level, and not only the project leader.

5.5 Observations of the Experimenters

Regardless of the team sentiment level, the study showed that awareness of a positive way of communicating can be increased with the help of such a tool. We observed an improved self-reflection and the careful selection of the wording in text messages. As soon as the participants perceived some kind of familiarity with the tool, they started to write messages as positive or as negative as possible in order to test the tool. Afterwards, they started adjusting their wording in order to be at least neutral, but not negative – also when trying to write some negative message in the fictive scenario.

6 Discussion

In the following, we discuss and interpret our results as well as threats to validity.

6.1 Answering the Research Questions

Based on our results, we can answer the research questions as follows:

Answer to RQ1: Using our rudimental and prototypical concept consisting of a keylogger and a joint sentiment analysis, it is possible to integrate realtime sentiment analysis in the daily lives of software developers. However, in order to ease the use of the concept, an integration of the concept into different platforms (such as Microsoft Teams or Skype) would be helpful. Nevertheless, a real use in practice is still missing. With our study, we just prove that the general concept works.

Answer to RQ2: The core idea is perceived helpful by our twelve study participants. However, the practitioners proposed different extensions to increase the helpfulness of the results, such as the integration into existing chat tools instead of the decentralized idea followed by our approach. These insights and the promising results motivate further research on realtime sentiment analysis.

Answer to RQ3: The participants also stated to be willing to share their aggregated scores to allow the calculation of an average team score (under the condition that they get to see the results). However, our study was conducted in an experimental setting. Thus, we got impressions from practitioners, but only reflecting on a hypothetical use in practice.

6.2 Interpretation

Summarizing, the results of our study motivate the further exploration of realtime sentiment analysis.

The results of our study indicate that realtime sentiment analysis is helpful and was well-accepted by the study participants. The median value was 4.5 (between agree and absolutely agree) when asked if such a tool could help in the future. The fact that this is not a "traditional", i.e., retrospective, sentiment analysis, but feedback when writing a message, means that realtime sentiment analysis does not contradict the current use of sentiment analysis. The established tools provide insights about sentiments by observing the textual communication of developers. The concept presented in this paper provides feedback on the sentiment of a self-written message with the option to adjust it appropriately before sending the message to the team. This helps to convey the perception a developer wants to convey accordingly and thus avoid possible misunderstandings. This means that, for example, developers can adjust their message from negative or neutral to positive if the message is not meant to have such polarity. On the other hand, it is possible to change a positive message to neutral, if this effect is desired, for example, due to poor results in the current development or inadequacy of positive mood due to other social factors.

Regarding the combination of realtime and "typical" sentiment analysis, there is the threat that messages are changed after the feedback of the tool, e.g., from negative towards positive, although the negative wording reflects the actually existing emotional mood of the sender. Regarding the reduced risk of negative mood in the team due to a too rough communication, this is the desired outcome. But teams need to be aware that a retrospective sentiment analysis would then not adequately reflect the real team mood. This means that the messages sent are somewhat manipulated and introduce a bias. Thus, sentiment analysis tools applied to written communication might not measure the true sentiment in this case. However, a combination between realtime and retrospective sentiment analysis is still useful to have more data to get closer to the real sentiment of a developer team. This was also requested by our study participants. In addition, the analysis of what has already been written can also have a different focus, e.g. instead of sentiment in the actual sense, it can be more of a content-based analysis, e.g. whether negative or positive events are mentioned. This also goes along with Graziotin et al. [5] results, which show that satisfied developers are better at solving problems. So if many negative events were mentioned in the communication, this can be an indication of possibly dissatisfied developers. This means that our feedback tool analyzes how a message might be perceived by the

recipient, while another tool analyzes the communication, focusing on how it was communicated. Nevertheless, these ideas require future research.

However, the security concerns mentioned by the participants must not be ignored. Therefore, a company or project leader should be very transparent with the tool (e.g., how it works and determines the sentiment) as well as with the analysis of the messages in order to avoid that the developers feel observed or do not feel comfortable writing messages. It also makes sense to regularly measure the mood and emotional state of developers to see whether such a feedback tool has a negative impact on the developers or not.

6.3 Threats to Validity

In the following, we present threats to validity according to our paper. We categorize the threats according to Wohlin et al. [32] as conclusion, internal, construct, and external validity.

Our study had only 12 participants (construct validity), and not all of them were pure developers (external validity). Consequently, the results must not be overinterpreted. We gained insights from twelve practitioners that imagined using such a tool in their daily lives. They answered the questions based on their experiences. However, already the study with such a small sample size provided interesting insights that motivate further research. As a first proof of concept, in our opinion, such a small study suffices. Nevertheless, future studies are required to strengthen the results.

The study was conducted in a fictive situation (construct and external validity) which limits the generalizability of our results.

The perceived accuracy of the realtime feedback tool offers potential for improvement, partially due to the lack of recognition of sarcasm and irony. These statements were mostly classified as neutral (internal and construct validity). However, as far as we know, this issue has not been resolved in research, but has been frequently identified as a problem [20,28].

When developers know that what they write is analyzed by a tool, this can influence their behavior. They would not write certain things or think about what they write before writing and sending it (external validity). However, this was not a threat of our study, but of the generalizability of the results. In this controlled settings, participants reported on a potential usefulness and it was not an "observation" of their behavior and what they write. They were just neutrally asked to try the tool. However, the fact of observation and monitoring will influence the use of the tool in reality.

Although all written communication is considered in the concept by a keylogger, written communication only accounts for 42% according to the surveys (conclusion validity). This can also include communication beyond the computer, e.g. via smartphone. Meetings, for example, are often also a component of communication. However, our concept covers a large part of the communication. And often misunderstandings and therefore problems and bad moods arise from written communication, where sender and receiver do not see each other and do not hear each other's voice.

In the study, only SentiStrength (for English and German) was used as a sentiment analysis tool (construct and internal validity). However, the tool was supplemented by further data analysis using emoticons and a self-created lexicon by the study participants. Nevertheless, the accuracy of the analysis results strongly depends on the selected analysis techniques. This accuracy, in turn, has an influence on the perceived helpfulness of the tool, because unreliable results would be neglected and will not provide any benefit for a hypothetical use in practice.

Open coding of the survey's responses to open questions was conducted by only one author (construct validity). However, the categories were reviewed by the other authors to increase the accuracy of the results.

Summarizing, these threats point to weaknesses of the reliability and the validity of our study's results. Future research is required to solve these issues, to provide deeper insights, and to strengthen the results. In particular, a real case study in practice is required to provide insights on the real usefulness and not only on the potential usefulness.

6.4 Future Work

The threats presented before and the results of our study point to potential for further research. For our study we used a keylogger to get access to the written message. In future work it is possible to develop a framework which can get access to other services like Microsoft Teams or Telegram. With this framework it should be possible to also handle emoticons which are integrated in the operated system and to also analyze reactions to specific messages. In our study we used only text-based emoticons. Operating systems have much more smileys with different meanings which can be analyzed. It is also possible that we receive better accuracy when we do not use lexicon-based sentiment analysis but a machine learning based algorithm. In the SLR from Obaidi and Klünder [28], we see that machine learning sentiment analysis offers a better performance, so we get better results.

To show the sentiments we present a prototype in Sect. 3 and there are much more possibilities to integrate the sentiment in other messengers. One of them could be a, e.g., highlighting-system like a spell checker. Sentiment words with positive meaning could be green and negative ones red.

Some of our participants wish to perform a long-time study to evaluate if the usage behavior is changing over time. In the future it is useful to conduct such a study. Furthermore, it can also be investigated to what extent the software has an influence on the users over time. For this purpose, interview studies or surveys on various aspects such as behavior, mood, etc. should be carried out before, during, and after the study.

Another point was that some users reported on being disappointed when writing a long message and only getting back an "okay". In future works it is possible to evaluate if an assistant can give suggestions for improvement e.g., "You got a long message. Don't you want to write more words in your response?" and if it is useful while writing messages. Another point to increase the effectiveness of

our prototype is that the sentiment analysis can (and should) also be improved, because we analyze only the text which is written but without any context analysis. If someone is writing ironically it is likely that the sentiment detection is wrong. Also, short messages without context could be wrong detected.

In order for the prototype to be used in a real scenario, certain aspects such as practicality must be taken into account. This may include ease of installation (or installation instructions), maintenance, and support for the software. This may include adapting the tool to the particular developer team. Communication between developer friends may be different than between the developer and the project manager.

The suggestions provided by our participants also point to the idea of combining realtime with retrospective sentiment analysis. As discussed before, this combination has potential, but requires a profound analysis of the real usefulness, as both analysis methods analyze the same aspect (the mood in the team) from different viewpoints and with different goals. Nevertheless, the combination might provide more sound insights than each of the methods (realtime and retrospective sentiment analysis) in a stand-alone setting.

7 Conclusion

As there is an increasing interest on analyzing social aspects in development teams, sentiment analysis is widely used in research to get an overview of what is going on in a development team. However, in order to provide the possibility for adjustments and interventions, some kind of realtime sentiment analysis is required.

Our concept for realtime sentiment analysis offers this opportunity by analyzing messages before they are sent. This avoids sending messages that raise some negative feeling with the receiver and increases the awareness of how some message might be perceived by others.

We evaluated our prototypically implemented concept in an experimental setting with twelve participants from industry. Despite the small sample size and the preliminary nature of this study, we observe that participants expect the idea of realtime sentiment analysis to be helpful in industry. In addition, they also stated to be willing to share their data to calculate the mood aggregated on team-level. Nevertheless, they provided ideas on how to extend the concept of realtime sentiment analysis pointed to already established approaches when doing retrospective sentiment analysis.

Future research will focus on the suggestions of the participants and requires a case study in industry in order to prove the usefulness in a real setting.

Acknowledgment. This research was funded by the Leibniz University Hannover as a Leibniz Young Investigator Grant (Project *ComContA*, Project Number *85430128*, 2020–2022).

References

1. Ahmed, T., Bosu, A., Iqbal, A., Rahimi, S.: SentiCR: a customized sentiment analysis tool for code review interactions. In: 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 106–111 (2017). https://doi.org/10.1109/ASE.2017.8115623
2. Cabrera-Diego, L.A., Bessis, N., Korkontzelos, I.: Classifying emotions in stack overflow and JIRA using a multi-label approach. Knowl.-Based Syst. **195**, 105633 (2020). https://doi.org/10.1016/j.knosys.2020.105633
3. Calefato, F., Lanubile, F., Maiorano, F., Novielli, N.: Sentiment polarity detection for software development. Empir. Softw. Eng. **23**(3), 1352–1382 (2017). https://doi.org/10.1007/s10664-017-9546-9
4. De Choudhury, M., Counts, S.: Understanding affect in the workplace via social media, pp. 303–316. Association for Computing Machinery, NY (2013)
5. Graziotin, D., Wang, X., Abrahamsson, P.: Happy software developers solve problems better: psychological measurements in empirical software engineering. PeerJ **2**, e289 (2014). https://doi.org/10.7717/peerj.289
6. Graziotin, D., Wang, X., Abrahamsson, P.: Do feelings matter? On the correlation of affects and the self-assessed productivity in software engineering. J. Softw.: Evol. Process **27**(7), 467–487 (2015). https://doi.org/10.1002/smr.1673
7. Guzman, E., Azócar, D., Li, Y.: Sentiment analysis of commit comments in GitHub: an empirical study. In: Kim, S., Pinzger, M., Devanbu, P. (eds.) 11th Working Conference on Mining Software Repositories. 31 May–1 June 2014, Hyderabad, pp. 352–355. ACM (2014). https://doi.org/10.1145/2597073.2597118
8. Guzman, E., Bruegge, B.: Towards emotional awareness in software development teams. In: Meyer, B., Mezini, M., Baresi, L. (eds.) 2013 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE). 18–26 August 2013, Saint Petersburg, Russia, pp. 671–674. ESEC/FSE 2013, ACM (2013). https://doi.org/10.1145/2491411.2494578
9. Guzman, E., Maalej, W.: How do users like this feature? A fine grained sentiment analysis of app reviews. In: 2014 IEEE 22nd International Requirements Engineering Conference (RE), pp. 153–162 (2014). https://doi.org/10.1109/RE.2014.6912257
10. Herrmann, M., Klünder, J.: From textual to verbal communication: towards applying sentiment analysis to a software project meeting. In: 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW), pp. 371–376 (2021). https://doi.org/10.1109/REW53955.2021.00065
11. Herrmann, M., Obaidi, M., Klünder, J.: Senti-analyzer: joint sentiment analysis for text-based and verbal communication in software projects. Tech. rep. (2022). https://arxiv.org/abs/2206.10993
12. Imtiaz, N., Middleton, J., Girouard, P., Murphy-Hill, E.: Sentiment and politeness analysis tools on developer discussions are unreliable, but so are people. In: Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering, pp. 55–61. SEmotion 2018, Association for Computing Machinery, NY (2018). https://doi.org/10.1145/3194932.3194938
13. Islam, M.R., Zibran, M.F.: Leveraging automated sentiment analysis in software engineering. In: 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), pp. 203–214 (2017). https://doi.org/10.1109/MSR.2017.9

14. Islam, M.R., Zibran, M.F.: DEVA: sensing emotions in the valence arousal space in software engineering text. In: Haddad, H.M., Wainwright, R.L., Chbeir, R. (eds.) Applied Computing 2018, pp. 1536–1543. Association for Computing Machinery Inc. (ACM), NY (2018). https://doi.org/10.1145/3167132.3167296
15. Islam, M.R., Zibran, M.F.: SentiStrength-SE: exploiting domain specificity for improved sentiment analysis in software engineering text. J. Syst. Softw. **145**, 125–146 (2018). https://doi.org/10.1016/j.jss.2018.08.030
16. Klünder, J., Horstmann, J., Karras, O.: Identifying the mood of a software development team by analyzing text-based communication in chats with machine learning. In: Bernhaupt, R., Ardito, C., Sauer, S. (eds.) HCSE 2020. LNCS, vol. 12481, pp. 133–151. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64266-2_8
17. Klünder, J., Schneider, K., Kortum, F., Straube, J., Handke, L., Kauffeld, S.: Communication in teams - an expression of social conflicts. In: Bogdan, C., et al. (eds.) HESSD/HCSE -2016. LNCS, vol. 9856, pp. 111–129. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44902-9_8
18. Kortum, F., Klünder, J., Schneider, K.: Don't underestimate the human factors! exploring team communication effects. In: Felderer, M., Méndez Fernández, D., Turhan, B., Kalinowski, M., Sarro, F., Winkler, D. (eds.) PROFES 2017. LNCS, vol. 10611, pp. 457–469. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69926-4_36
19. Kortum, F., Klünder, J., Schneider, K.: Behavior-driven dynamics in agile development: the effect of fast feedback on teams. In: 2019 IEEE/ACM International Conference on Software and System Processes (ICSSP), pp. 34–43. IEEE (2019)
20. Lin, B., Cassee, N., Serebrenik, A., Bavota, G., Novielli, N., Lanza, M.: Opinion mining for software development: a systematic literature review. ACM Trans. Softw. Eng. Methodol. **31**(3), 1–41 (2022). https://doi.org/10.1145/3490388
21. Lin, B., Zampetti, F., Bavota, G., Di Penta, M., Lanza, M., Oliveto, R.: Sentiment analysis for software engineering: how far can we go? In: Proceedings of the 40th International Conference on Software Engineering, pp. 94–104. ICSE 2018, Association for Computing Machinery, NY (2018). https://doi.org/10.1145/3180155.3180195
22. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach (2019)
23. Freira, M., Caetano, J., Oliveira, J., Marques-Neto, H.: Analyzing the impact of feedback in GitHub on the software developer's mood (2018). https://doi.org/10.18293/SEKE2018-153
24. Graziotin, D., Wang, X., Abrahamsson, P.: How do you feel, developer? An explanatory theory of the impact of affects on programming performance. PeerJ Comput. Sci. **1**, e18 (2015)
25. Novielli, N., Girardi, D., Lanubile, F.: A benchmark study on sentiment analysis for software engineering research. In: 2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR), pp. 364–375 (2018)
26. Novielli, N., Calefato, F., Dongiovanni, D., Girardi, D., Lanubile, F.: Can we use SE-specific sentiment analysis tools in a cross-platform setting? pp. 158–168. Association for Computing Machinery, NY (2020)
27. Novielli, N., Calefato, F., Lanubile, F., Serebrenik, A.: Assessment of off-the-shelf SE-specific sentiment analysis tools: an extended replication study. Empir. Softw. Eng. **26**(4), 1–29 (2021). https://doi.org/10.1007/s10664-021-09960-w
28. Obaidi, M., Klünder, J.: Development and application of sentiment analysis tools in software engineering: a systematic literature review. In: Evaluation and Assessment in Software Engineering, pp. 80–89. EASE 2021, Association for Computing Machinery, NY (2021). https://doi.org/10.1145/3463274.3463328

29. Schroth, L.: Konzept für eine präventive Stimmungsanalyse von Nachrichten in Software-Entwicklungsteams (concept for a preventive sentiment analysis of messages in software development teams) (2022)
30. Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment strength detection for the social web. J. Am. Soc. Inform. Sci. Technol. **63**(1), 163–173 (2012). https://doi.org/10.1002/asi.21662
31. Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A.: Sentiment strength detection in short informal text. J. Am. Soc. Inform. Sci. Technol. **61**(12), 2544–2558 (2010). https://doi.org/10.1002/asi.21416
32. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-29044-2
33. Zhang, T., Xu, B., Thung, F., Haryono, S.A., Lo, D., Jiang, L.: Sentiment analysis for software engineering: how far can pre-trained transformer models go? In: 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 70–80 (2020). https://doi.org/10.1109/ICSME46990.2020.00017