Identifying the Mood of a Software Development Team by Analyzing Text-Based Communication in Chats with Machine Learning

Jil Klünder, Julian Horstmann, Oliver Karras

Identifying the Mood of a Software Development Team by Analyzing Text-Based Communication in Chats with Machine Learning

Jil Klünder[0000−0001−7674−2930], Julian Horstmann, and Oliver Karras[0000−0001−5336−6899]

Leibniz University Hannover, Software Engineering Group, Hannover, Germany
{jil.kluender | oliver.karras}@inf.uni-hannover.de,
julian.horstmann@se.uni-hannover.de

Abstract. Software development encompasses many collaborative tasks in which usually several persons are involved. Close collaboration and the synchronization of different members of the development team require effective communication. One established communication channel are meetings which are, however, often not as effective as expected. Several approaches already focused on the analysis of meetings to determine the reasons for inefficiency and dissatisfying meeting outcomes. In addition to meetings, text-based communication channels such as chats and e-mails are frequently used in development teams. Communication via these channels requires a similar appropriate behavior as in meetings to achieve a satisfying and expedient collaboration. However, these channels have not yet been extensively examined in research.
In this paper, we present an approach for analyzing interpersonal behavior in text-based communication concerning the conversational tone, the familiarity of sender and receiver, the sender's emotionality, and the appropriateness of the used language. We evaluate our approach in an industrial case study based on 1947 messages sent in a group chat in Zulip over 5.5 months. Using our approach, it was possible to automatically classify written sentences as positive, neutral, or negative with an average accuracy of 62.97% compared to human ratings. Despite this coarse-grained classification, it is possible to gain an overall picture of the adequacy of the textual communication and tendencies in the group mood.

Keywords: Communication · development teams · software projects · human aspects · interpersonal behavior

1 Introduction

Due to the increasing complexity of software, most software projects require some kind of teamwork [17]. Having a team working on a project requires coordination and an adequate collaboration [6, 17], for example, appropriate requirements communication between the development team and the customer. To succeed

with the project, the team and the customer must share the same vision [4]. Otherwise, the team cannot develop a software satisfying the customer [2].

An adequate collaboration requires knowledge and information sharing to have a successful project closure [21,24]. Lost or insufficiently shared information can cause – in the worst case – project failure, e.g., due to missing functionality of the final software product. Mitigating this risk requires a sufficient amount of communication during the whole development process [21]. This communication can take place, e.g., in meetings, via e-mail, or during phone calls [16].

As meetings enable team members to share a lot of information with many team members in a short time, they are an established medium in the development process [23, 26]. However, inappropriate behavior and interactions in meetings decrease the success of a meeting and the participants' satisfaction afterwards [24,26]. This, in turn, has an influence on the project and the collaboration. To avoid inappropriate behavior in meetings, interaction analyses are an established medium in psychology [13, 19] and gain increasing attention in software engineering [15, 24, 26].

In addition to the increasing complexity of software projects, the share of globally distributed projects is high [18] and complicates a close collaboration [27,31]. In case of regionally or globally distributed projects, it is difficult to have meetings regularly [23]. Virtual meetings are a possibility, which is, however, influenced by the requirements for technical equipment (including bandwidth) and difficulties caused by different time zones. Therefore, indirect communication using digital communication channels such as e-mails or instant messenger is widely used in software projects [16].

Problem Statement. According to Schneider et al. [26] and Kauffeld et al. [14], a single person participating in a meeting can influence the mood of all other participants – both positively or negatively. This, in turn, influences the developers' productivity [8] and has several other consequences for the project [7]. Therefore, interaction analyses in meetings take into account the amount of positive, i.e., good and appropriate, as well as negative, i.e., bad and inappropriate, behavior during the meetings [13, 15]. Since the frequency and duration of meetings tend to decrease with project progress, whereas the use of other communication channels increases [16], likely, the communication behavior in text messages can also influence team satisfaction and, thus, motivation and project progress.

Objective. In this paper, *we want to analyze text-based communication, for example in e-mails or chats, in development teams with respect to its emotionality to detect development phases where the group mood is rather negative.* The emotionality of text-based communication is for example affected by the language used, the frequency, the length of the messages, the formality of the communication, the use of emoticons, and the time until the receiver replies to the message. In particular, we want to answer the following research question:

Research Question 1

How can text-based communication in development teams be holistically analyzed to derive information on the mood in the team?

Contribution. We present the current state of our approach classifying written messages as *positive*, *negative*, or *neutral* based on the sensitivity and the formality of the used words. We evaluate the approach in a case study in industry based on 1947 messages in a group chat. The results show that our tool classifies single sentences as *positive*, *negative*, or *neutral* with an average accuracy of 62.97%. When refining the analysis techniques it is possible to increase this number of correctly identified sentences either to shed light on the overall mood transported in messages or to analyze the mood in the development team based on the messages to allow interventions in case of a very dissatisfied team.

Outline. The rest of the paper is structured as follows: In Section 2, we present related work. Section 3 summarizes the concept and the approach followed in this paper. We evaluate the approach in Section 4 and present the results of the application in industry in Section 5 which we discuss in Section 6. We conclude the paper in Section 7.

2 Related Work

Analyzing communication behavior is not new in the area of Software Engineering. McChesney and Gallagher [22] analyze both communication and coordination in software projects. Herbsleb and Mockus [9] analyze differences in communication behavior of distributed and co-located teams. Klünder et al. [16] analyze team meetings, their frequency and duration over time in software projects. On a more fine-grained level, Schneider et al. [26] analyze interactions in team meetings of development teams. All these analyses require manual effort.

However, there are some approaches to support the analysis of communication of development teams by tools. Most existing approaches focus on meetings. Gall and Berenbach [5] present a framework to record elicitation meetings and automatically save information given by stakeholders. Shakeri et al. [1] also support the analysis of elicitation meetings. Their tool automatically collects knowledge that is important to understand the requirements. This approach is mainly based on written documentation and allows a content-related analysis. However, it has not yet been applied to written communication in text messages of development teams.

Sentiment analysis, i.e., the analysis of textual language aiming at identifying the author's mood, is also not new. Jongeling et al. [11] compare different tools for sentiment analysis in the Software Engineering domain and compare the tools' results to human evaluators. According to their results, different tools can produce contradictory results. Islam and Zibran [10] also analyze and compare the results of different tools used for sentiment analysis to understand their low accuracy. They present an improved version of one of the tools adjusted to development teams. The inaccuracy of tools can be partially explained by training the used classifiers on data sets which are not related to the Software Engineering domain and hence do not consider domain-typical language and knowledge. Lin et al. [20] trained an already existing tool for sentiment analysis using 40000 manually labeled sentences or words from Stack Overflow. Calefato

et al. [3] present Senti4SD which is a classifier adjusted to development teams. The classifier was trained using communication of developers on Stack Overflow. Jurado and Rodriguez [12] analyze text in issues and tickets with sentiment analysis to extend the possibilities to investigate the development process.

In this paper, we use sentiment analysis to analyze text-based communication of developers on team level, which has, to the best of our knowledge, not been done before.

3 General Research Approach

To achieve our research goal, we combine sentiment analysis with natural language processing and machine learning to classify text-based communication. We developed an approach to classify text-based communication as *positive, neutral* or *negative* considering the emotion transported in the message. In the following, we give an overview of the process presented in Fig. 1. An exemplary application of our approach is presented in Section 4.

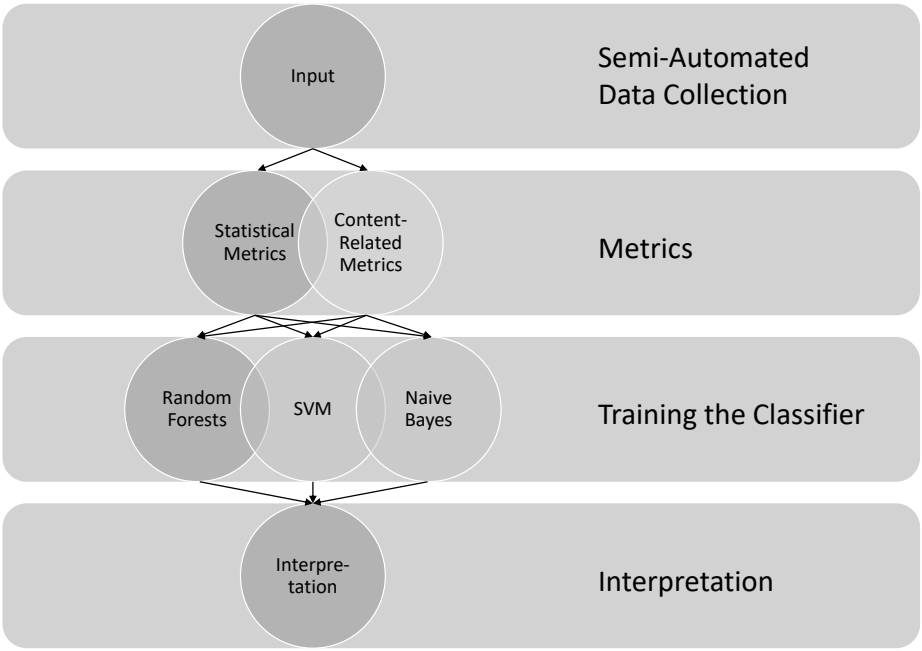

Fig. 1. Overview of the process to classify messages

Our approach consists of the four steps summarized in Fig. 1. It starts with a semi-automated data collection by crawling the respective communication channel. Afterwards, different metrics are calculated for the text messages to extract

relevant characteristics of the messages to allow conclusions regarding the emotionality of the communication. These metrics are used by a trained classifier to assign one of the classes *positive, neutral*, or *negative* to each of the sentences. This classification allows an evaluation and interpretation of the whole communication, e.g., on a daily basis.

3.1 Step 1: Data Collection

Text-based communication can be found in different sources, including mailing lists, instant messengers such as Slack[1], Skype[2], or Zulip[3], and in e-mails. To analyze the communication on group level, it is most suitable to use some kind of group chat or mailing lists several team members have access to and use it to discuss team-internal project-related issues. However, the process is applicable to any kind of text-based communication, including bi-directional information exchange[4].

The semi-automated data collection strongly depends on the communication channel under consideration. Exporting text messages in Skype differs from a data export in e-mails. At the moment, our approach supports the semi-automated data collection for Zulip, which will be described in more detail in Section 4 as part of the application in industry.

3.2 Step 2: Metrics

Metrics are required to extract relevant characteristics of text-based communication. The identification of relevant metrics is difficult as natural language is not unique and can be interpreted in completely different ways [30]. Depending on the actual goal of the communication analysis, one may choose different metrics. In the following, we present some exemplary metrics and the rationale of why we consider them useful for emotional analysis of text-based communication.

Statistical metrics analyze the text messages from a quantitative viewpoint, for example by calculating the *length of each message* or the *average length of the used words*. The length of the words and the message allow conclusions for the formality of the communication. Consider the following situation:

[1] `https://slack.com/`
[2] `https://www.skype.com/`
[3] `https://zulipchat.com/`
[4] Note that the analysis of bi-directional communication is questionable due to privacy concerns and personal messages in private chats.

Example

Paul has a problem and writes a message with 100 words in the group chat. In his text, he explains the problem in detail so that everybody who reads the message has a clear idea of what information he needs to solve the problem. In the end, he proposes a very time-consuming alternative if he does not get the required information. And the only answer he receives comes from Anton: "sounds good".

This huge difference in the length of the messages allows several conclusions. First, it raises the impression that nobody but Anton cares about Paul's problem. Second, as Anton's reaction is short, he may not even have read the whole text, or is at least not interested in supporting Paul. Of course, one can also interpret the short message differently. But this is one scenario, where a short message can lead to demotivation and team-internal problems. And it is not clear how Paul interprets this answer.

Besides the lengths of messages and words, counting adjectives, emoticons and the number of punctuation characters is also useful. The use of emoticons indicates familiarity in the team. In formal communication, emoticons will only be partially used, if at all. The number of punctuation characters is difficult to interpret. On the one hand, the use of commas and the like imply a formal message, but it can also indicate a huge amount of emoticons such as ":-)". Nonetheless, in conjunction with a detection of smileys using punctuation characters can help to analyze the formality of the message.

Content-Related Metrics The statistical metrics do not allow conclusions on the emotionality of the messages which is the main topic of this paper. Therefore, the content of text-based communication has also to be taken into account. To analyze the communication on content-level, profound knowledge on a typical structure of the language, the so-called *part of speech*, is necessary. Using natural language processing, it is possible to analyze words in relation to their position in the sentence. This allows considering not only the word A itself but also other words that may influence the interpretation of the word A.

Emoticons also raise feelings, which may differ among the receiver of a message [29]. Wang and Castanon [29] investigated the use of emoticons and the intention of the use. According to their results, it is not always possible to assign exactly one of the classes *positive, neutral*, or *negative* to the emoticon [29]. Therefore, they present a list of emoticons together with a probability that the emoticon belongs to the respective class. For example, the emoticon ":D" is interpreted positively with a probability of 90%. In the approach presented in this paper, we assign the class with the highest probability to the respective emoticon. However, at the moment, we do not consider neutral emoticons. This will be part of future research.

Comparable to the feelings raised when seeing emoticons, words also have some kind of emotional shade. To identify the emotional shade of the words, we

Table 1. Examples for the emotional shades [28]. The number in brackets, if available, represent the score [25].

positive	neutral	negative
agree (0.0040)	objectively	arbitrary (-0.3481)
convinced (0.2381)	fully	confused
innovation (0.0040)	thought	deficient (-0.4535)
non-violence	argumentation	autocrat

used two databases [5] summarizing words and their emotionality [25, 28]. The database provided by Waltinger [28] assigns each word to one of the classes *positive (+1), neutral (0)*, or *negative (-1)*, whereas Remus et al.'s [25] database assigns a value ranging from *-1 (negative)* to *1 (positive)* to each word. We use both databases to increase the number of words and to aggregate the results. Examples from the first database can be found in Table 1. If possible (i.e., if the word is also contained in the second database), we present the concrete value in brackets.

In addition, we use the *CountVectorizer* and the *TF-IDF Vectorizer* to analyze the relevance of words. For example, the *TF-IDF Vectorizer* calculates the term frequency of the word in the message (TF) and the inverse document frequency (IDF) considering all messages. This helps to detect relevant words which only appear a few times in the text.

We also take the *formality* of the language into account. However, the detection of formality is language-specific (e.g., in Spanish and German (and other languages) there are specific forms to address someone formally).

3.3 Step 3: Training the Classifier

In the next step, we train the classifier to achieve good classification results. We use machine learning techniques to train the classifier, including *Random Forests,* a *Support Vector Machine*, and *Naive Bayes*. To increase the accuracy of our model, we combine the results of the three methods using a *Voting Classifier* choosing the class for a sentence that was forecasted by the majority of the three approaches.

A *Random Forest* starts with randomly creating decision trees. Each of the trees classifies the sentence. In the end, the algorithm chooses the class that is most often chosen by the trees. An schematic visualization is presented in Fig. 2.

A *Support Vector Machine* separates the multi-dimensional feature space. Consider the 2-dimensional example in Fig. 3. The gray dots represent data points, i.e., messages, labeled as *neutral*, and the black dots represent data points classified as *negative*. The Support Vector Machine separates this two-dimensional space using a linear function. Therefore, the gray dot circled with black dots would be classified as negative even if the sentence is neutral.

[5] Note that both databases are based on the German language as we performed our application in industry (see Section 4) in a German-speaking company.

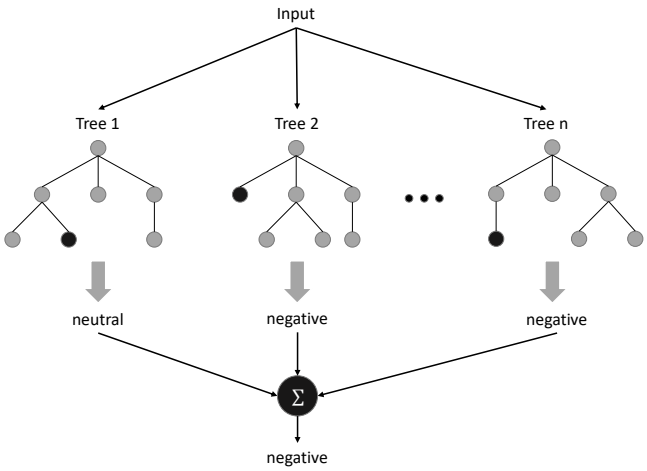

Fig. 2. Schematic Visualization of the Random Forest

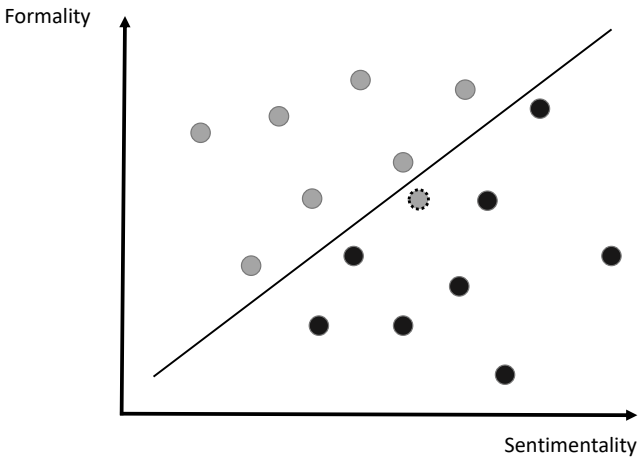

Fig. 3. Example for a two-dimensional classification by a Support Vector Machine

The *Naive Bayes Classifier* uses probability functions indicating whether a data point in the feature space belongs to a specific class or not. As the density functions of the probability are unknown, this procedure allows only for an approximation.

As all of the three approaches have their strengths and weaknesses, we decide to combine their results using a *voting classifier*. This classifier identifies the class which was chosen by the majority of the approaches. This way, the approach is more robust against outliers and we can reduce the influence of noise in the data.

The training of the classifier can be done by dividing the data set into training and test data. The training data is then used to derive heuristics indicating that a specific type of sentence belongs to a specific class (i.e., to train the classifier) and the test data is used to check the accuracy of the classifier.

To optimize the forecast, we use an evolutionary algorithm. This algorithm uses a fitness function representing the goodness of the current solution, i.e., the classifier. In our case, the goodness is defined via the average accuracy of the classification. We repeat the learning process 20 times with different test and training sets (in a ratio of 10:90). This allows cross-validation of the classifier.

3.4 Step 4: Interpretation of the results

The interpretation of the results is mainly part of future work. However, to summarize and visualize the emotionality of messages over time, we map the three classes *positive, neutral*, and *negative* to the integer values *+1, 0,* and *−1*. This allows to calculate mean values, for example of all messages sent on a specific day, and analyses over time.

4 Application in Industry

To show the applicability of our approach, we apply it to an industrial software project. Figure 4 shows the four steps of our general research process with its concrete instantiation in the case study which required a step for the data preprocessing. In the following, we present our proceeding when applying the approach in detail.

4.1 Data Collection

We evaluated our approach in industry. However, due to privacy concerns, we cannot provide profound information on the company. We call the company *ZETA*. *ZETA* is specialized on software and consulting. There is one in-house software development team called *Team MY*. Before running the case study in the company, each team member of *Team MY* received a transparency letter describing the overall concept of our approach as well as a detailed description of the data analysis of communication data of the whole team. After a reflection period of two weeks to communicate concerns, ideas, or disagreement, we were allowed to start the case study since none of the team members disagreed.

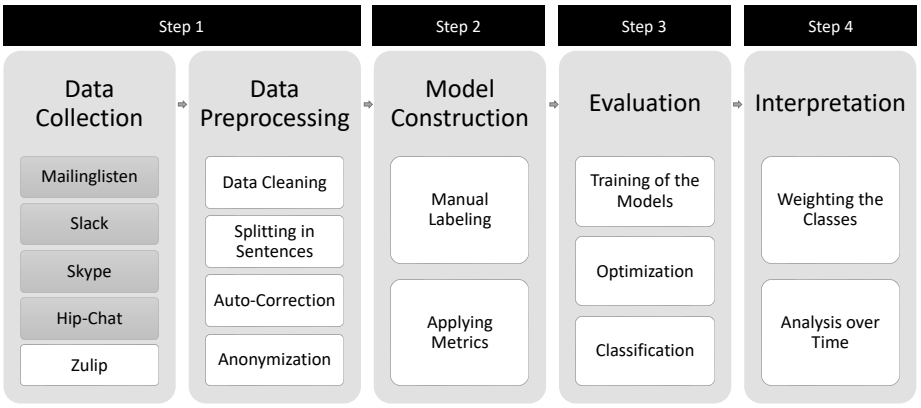

Fig. 4. Overview of the application in industry

In total, about 80 developers worked on the software project we use to prove the applicability of our approach in industry. These developers work distributed in one country. The main communication tool in the software project is *Zulip* which is a chat tool for teams[6]. The users subscribe to so-called *streams*. Each stream has a *topic* that organizes conversations in the stream. This helps to cluster messages also after hours of silence on a respective topic. The user can decide on his access to the streams. This reduces the risk of information overload with irrelevant information. Besides Zulip, the team also uses (if possible) face-to-face communication and meetings, as well as phone calls, e-mails, and a chat tool. However, Zulip is the official tool that should be used for communication.

The company granted us access to five streams of *Team MY* which were exported using the REST-API for further processing. We exported all messages that have been written between Feb 11, 2019 and Jul, 24 2019, resulting in 1947 messages consisting of 7070 sentences. In total, 65 developers actively participated in the communication in at least one of the five streams. Note that we consider a person to actively participate if and only if she wrote at least one message.

4.2 Data Processing

To apply our research process, it was necessary to preprocess the data. The data processing consisted of several steps, starting with the data preparation. This includes (1) data cleaning to remove irrelevant data such as source code which cannot be processed, (2) cutting the messages into sentences, (3) the auto-correction, and (4) anonymization of the messages.

(1) The data export contains several messages respectively strings that cannot be processed. This includes symbols for text highlighting such as asterisks

[6] More information on Zulip can be found at `https://zulipchat.com/`.

(for bold text) or low lines (for italic text) as well as links and source code. These parts have to be removed. This step was, in our case, done manually. However, we plan to automatize this step in future research.

(2) To increase the level of detail of the analysis, the unit of analysis is a single sentence. Therefore, messages which consist of more than one sentence have to be split into single sentences. As start and end of a sentence cannot always be derived by the punctuation, we use the Python package *spacy*. This package identifies sentences based on the structure of the language (and not solely on the punctuation). However, as *spacy* does not have an accuracy of 100%, we manually checked the results.

(3) Most metrics require the correct spelling of words. Otherwise, words may be identified as a different word mitigating the correctness of the classification. Therefore, we used the Python package *pyspellchecker* providing corrections for misspelled words.

(4) During the anonymization phase, we manually replaced all names and addresses (including links) by pseudonyms. This step was done by one author of this paper. Due to privacy concerns, only one researcher was allowed to process the raw data. Limitations caused by this fact will be discussed in Section 6.

4.3 Training the Model

In the next step, we trained the model to evaluate the accuracy of the classifier. The training consisted of two steps: (1) Labeling of existing sentences, and (2) training the classifier.

(1) We manually labeled the data by assigning one of the three classes *positive, neutral,* and *negative* to each of the sentences. Examples for words belonging to the respective class are presented in Table 2. The division in three classes is rather coarse-grained. Future research will focus on increasing the granularity of the results. Due to the privacy concerns of the company, this step was done by one researcher. Limitations caused by this fact will be discussed in Section 6.

(2) In the training phase, the metrics presented in Section 3.2 are calculated for each of the sentences. We calculated in total 5378 different metrics, mostly considering language-specific characteristics[7]. We chose this huge number of metrics as each metric which is not correlated to other metrics can increase the accuracy of the classifier. To analyze the correlation between the metrics, we calculated the correlation matrix (based on Pearson's r) for this specific data set.

[7] As the number of metrics is quite high, future research should focus on the selection of appropriate metrics.

Table 2. Exemplary emotions or types of words for each class

positive	neutral	negative
love	facts	fear
happiness	ambivalence	hate
euphoria	interest	anger
surprise	indifference	trouble
sympathy	apathy	regret

4.4 Evaluating the Model

We evaluate the model by applying the classifier to some exemplary sentences of our data set. We chose a ratio of 10% for the test data set and use the remaining 90% of the data set to train the model. The selection of the concrete test data was random. The learning process of the model is based on an evolutionary algorithm which identifies the best fitting model. In our case, we achieved the best model after 80 steps. The accuracy of the model is defined by the confusion matrix summarizing the results of the classification (comparison between forecast and as-is) as well as different key indicators such as precision, recall, and F1-score.

4.5 Interpretation of the results

To visualize the results, we present the development of the average emotional score of the messages over time. We assign values to each of the classes, namely *positive = +1*, *neutral = 0*, and *negative = -1*. This way, it is possible to calculate the average score per day, which we call the *emotionality score*. Observing this score over time can help to detect stressful phases. However, the interpretation of the emotionality score remains future work.

5 Results

We performed the steps described in Section 4. The manual data labeling resulted in 845 sentences classified as *positive*, 1856 sentences classified as *neutral*, and 1077 sentences classified as *negative*. This imbalance of data points can influence the classifier's accuracy as there are more negative sentences to train the classifier than positive. However, it is unlikely to find completely balanced data in industry. Future research should, hence, focus on training the model with huge data sets containing balanced data.

In a next step, we calculated the correlation matrix of the metrics presented in Fig. 5. Most of the metrics are only partially or not at all correlated. However, the metrics for the emotions (emoji_mean, emoji_min, and emoji_max) are highly correlated. In addition, we find a weak correlation between the need for corrections (AutoCorrectionRatio) and the formality of the communication (formality).

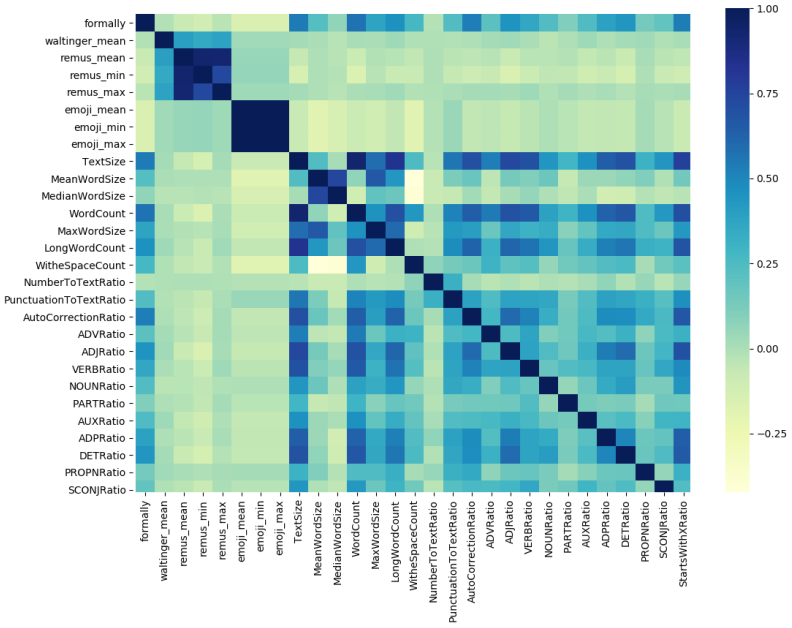

Fig. 5. Correlation matrix including 28 metrics

Table 3. Confusion matrix of the classifier for the test set

	positive	**neutral**	**negative**
positive	36	46	7
neutral	10	173	11
negative	2	64	29

As described above, we trained the model using the labeled data set. The classifier assigned one of the classes *positive, neutral,* and *negative* to the sentence under consideration. The evolutionary algorithm used to train the model achieved, in his best generation, an average accuracy of 58.5%. We chose the best model from this generation which achieved an average accuracy of 62.97%. To calculate the accuracy of our model, we applied an amount of randomly chosen 10% of the data set as test data to the model trained by the remaining 90% of the data set as training data. The results of the classification are presented in Table 3 as a confusion matrix. We see only 9 out of 378 confused classifications of positive and negative, i.e., it was most difficult to distinguish between positive and neutral (56 out of 378 confusions) respectively between negative and neutral (75 out of 378). Based on these numbers, it is possible to calculate other key indicators summarized in Table 4. In total, the class *positive* has the highest precision, whereas the F1-Score is best for the neutral class. Examples for a few classifications are presented in Table 5 presenting the manual classification by a person and the predicted classification by the model.

Table 4. Classification report for the test set

Class	Precision	Recall	F1-Score	Frequency
positive	0.75	0.40	0.53	89
neutral	0.61	0.89	0.73	194
negative	0.62	0.31	0.41	95

Table 5. Exemplary sentences and predictions from a human rater and the trained classifier

Id	Content	Human	Classifier
1	Yes, this was my mistake.	negative	neutral
2	Welcome in [[countryname]] :-) good decision	positive	positive
3	If this was not your mistake, it has to be fixed as follows:	neutral	neutral
4	Had understood you differently this morning.	negative	neutral
5	Then we agree.	positive	neutral
6	Well done!	positive	positive

At this point, we have evaluated the classifier leading to an accuracy of 62.97%. In the last step, we want to outline how the classification (either from a person or from the classifier) can be used to analyze emotions in the team. Presenting the shade of the emotions on a scale from -1 to 1 as described in Section 3.4 leads to the curve presented in Fig. 6. Note that we use the manually labeled data. However, this step can also be done using the results of the classifier (see Fig. 7). To derive the results in Fig. 6, we first calculate the average emotional score of the labeled data on a daily level. As evident from Fig 6, the

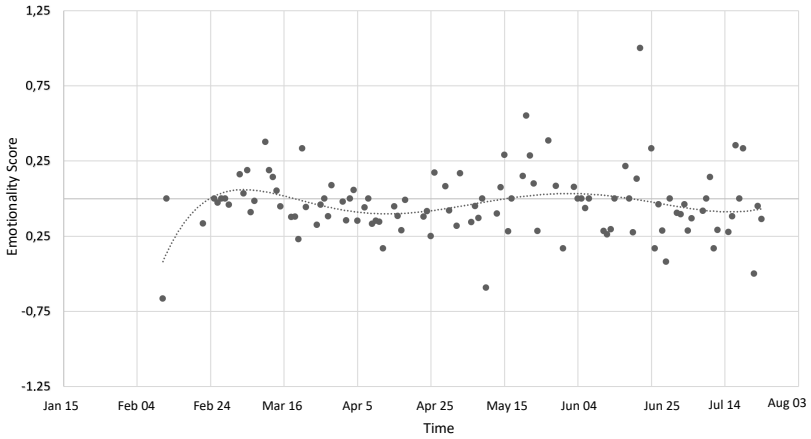

Fig. 6. Emotional niveau using the manually labeled data. The dotted curve presents the trendline.

mood is on average rather neutral. However, it ranges from positive to negative and vice versa. When presenting the results to the company, we figured out that the negative phases coincide with stressful phases in the project, e.g., due to deadlines. Comparing the trendline in Fig. 6 with the trendline in Fig. 7 based on the forecasted data shows very similar tendencies.

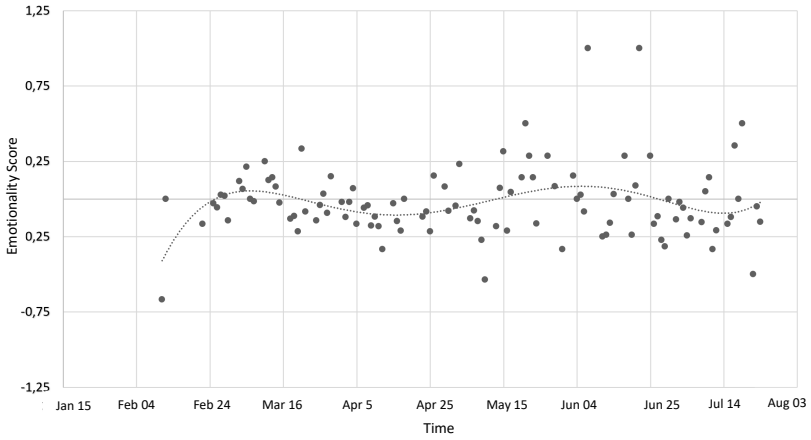

Fig. 7. Emotional niveau using the forecasted data. The dotted curve presents the trendline.

6 Discussion

In the application in industry, we analyzed text-based communication consisting of 1947 messages in a chat tool used by a development team in industry. We used several metrics ranging from the emotional shade of single words over the length of a sentence to the formality of the language. We combined three machine learning algorithms to classify each of the messages as *positive, neutral*, or *negative*.

Our approach achieved an accuracy of 62.97% which allows for improvement. However, the trained classifier was almost as good as a human rater who achieved an accuracy of 66% when coding 200 sentences twice with a temporal distance of one month.

We used the classification to aggregate the team mood on a daily level by calculating the average emotionality of all messages sent on a specific day. When interpreting the course of emotionality over time, we see an average neutral mood, with some tendencies towards positive as well as negative mood. However, one would expect to find such a rather balanced course in a professional work environment. Further possibilities to interpret the results will be subject to future research.

6.1 Threats to Validity

Our application in industry is subject to some threats to validity which we discuss now.

The *conclusion validity* is threatened by the choice of the data source. For example, communication in e-mails will differ from communication in group chats. We are aware that adjustments to the procedure are required, as the data collection and extraction strongly depends on the data source. To provide a holistic picture of the emotionality of the text-based communication in the team, we consider a huge set of metrics. To allow good learning of the model, we ensure that these metrics are only partially correlated and do not measure all the same characteristics of the communication.

Due to privacy concerns and legal restrictions from the company, only one author was allowed to handle the data. This threatens the *internal validity* since the data processing strongly depends on the subjective perception of one person. By following a structured proceeding, for example for the anonymization of the data, we tried to mitigate this threat. However, there was no possibility to review the labeling process. Therefore, the person who labeled the data labeled a randomly chosen set of 200 sentences a second time with a distance of one month. In 2 of 3 cases, the labels coincided, leading to an accuracy of 66%. This raises several questions and possibilities for future research which will be discussed in Section 6.3. However, the manual reference classification of the data influences the results as this defines whether the classification by the tool is correct or not.

The *construct validity* may be threatened due to the mono-operation and the mono-method bias. As we are aware of this fact, we do not draw any conclusions based on our results. We only see them as promising support to continue the

development of the approach. Further studies in different settings are required to mitigate this threat.

The *external validity* of our results is very limited. The concrete results of our analysis are correct for the team under consideration, and only reflect the time frame of our analysis. However, we do not want to draw conclusions or to generalize our results for other teams. The main objective of the application to an industrial project is to prove the applicability of our approach. If this is the case, we only conclude that our approach is applicable, but the results obtained from the study cannot be transferred to any other team. How to draw conclusions based on the results will be subject to future work (see Section 6.3).

6.2 Answering the Research Question

With the approach presented in this paper and its application in industry, we were able to show that the analysis of text-based communication is possible using a combination of natural language processing, sentiment analysis, and machine learning. By aggregating the emotionality of the sentences, e.g., daily, it is possible to derive information on the general mood in the team as well as the development over time. However, there are some open questions and there is potential for improvement to be addressed in future work.

6.3 Future Research

Even if the results of our approach underline its potential, it might (and should) be improved to address some issues that could not have been considered in the paper at hand. These issues will be addressed in future research.

(1) One problem is the rather low accuracy of the algorithm. However, given the interrater agreement of a human rater of 66% when labeling the data twice with a temporal distance of one month, the classifier is almost as good as the human rater. This result is near to perfect as a trained model cannot perform better than the labeling allows. Consequently, to improve the accuracy of our algorithm, we first have to find possibilities to identify clear characteristics of the messages in the respective class. This finding goes along with the findings of other authors [11].

(2) The context of the single sentence is currently not considered. This should be improved in future research. Depending on the context of previous messages, an answer can be differently interpreted. Therefore, the emotionality of the sentences before also needs to be considered for the forecast and probably can also improve the classification.

(3) At the moment, we do not provide any guidance for the interpretation of the results. To gain insights based on the results, we calculate average emotionality scores and present them on a daily level. This can help to detect stressful phases or phases where the team needs support, for example by an external coach due to unsolved team-internal conflicts. However, as part of future research, it is required to apply our approach to huge data sets and investigate correlations to other factors (e.g., deadlines, conflicts, high workloads, and context factors).

(4) The usefulness of the tool for development teams is not yet proven. This is mainly because we do not provide guidance to interpret the results. As soon as we finished the research described in (3), we can evaluate the usefulness of the insights by presenting them to the team.

7 Conclusion

As text-based communication is widely distributed in software development, but often not adequate, we strive towards an automated analysis of text-based communication in different channels. Our approach analyzes interpersonal behavior in text-based communication concerning the emotional shade of the communication, considering the conversational tone, the familiarity of sender and receiver, and the appropriateness of the used language. We prove the applicability of our approach in an industrial case study, where we got access to 1947 messages in Zulip, consisting of 7070 sentences. The results of our application in industry show that it is possible to correctly classify statements with an average accuracy of 62.97% which is as good as the rating of a human classifier. In this paper, we only classified the statements as *positive*, *neutral*, or *negative*. In future work, we want to make the results more fine-grained and also consider other information contained in the data, such as emoticons. In addition, we want to support the interpretation of the results, which is not yet part of the approach.

References

1. Abad, Z.S.H., Gervasi, V., Zowghi, D., Barker, K.: Elica: An automated tool for dynamic extraction of requirements relevant information. In: Proceedings of the 5th International Workshop on Artificial Intelligence for Requirements Engineering. IEEE (2018)
2. Bjarnason, E., Wnuk, K., Regnell, B.: Requirements are slipping through the gaps—a case study on causes & effects of communication gaps in large-scale software development. In: 2011 IEEE 19th international requirements engineering conference. pp. 37–46. IEEE (2011)
3. Calefato, F., Lanubile, F., Maiorano, F., Novielli, N.: Sentiment polarity detection for software development. Empirical Software Engineering **23**(3), 1352–1382 (2018)
4. Fricker, S.A., Grau, R., Zwingli, A.: Requirements engineering: best practice. In: Requirements Engineering for Digital Health, pp. 25–46. Springer (2015)
5. Gall, M., Berenbach, B.: Towards a framework for real time requirements elicitation. In: Proceedings of the 1st International Workshop on Multimedia Requirements Engineering. p. 4. IEEE (2006)
6. Ghosh, T., Yates, J., Orlikowski, W.: Using communication norms for coordination: Evidence from a distributed team. Proceedings of the 25th International Conference on Information Systems p. 10 (2004)
7. Graziotin, D., Fagerholm, F., Wang, X., Abrahamsson, P.: Consequences of unhappiness while developing software. In: Proceedings of the 2nd International Workshop on Emotion Awareness in Software Engineering. pp. 42–47. IEEE Press (2017)

8. Graziotin, D., Wang, X., Abrahamsson, P.: Are happy developers more productive? In: International Conference on Product Focused Software Process Improvement. pp. 50–64. Springer (2013)
9. Herbsleb, J.D., Mockus, A.: An empirical study of speed and communication in globally distributed software development. IEEE Transactions on Software Engineering **29**(6), 481–494 (2003)
10. 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. IEEE (2017)
11. Jongeling, R., Datta, S., Serebrenik, A.: Choosing your weapons: On sentiment analysis tools for software engineering research. In: 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 531–535. IEEE (2015)
12. Jurado, F., Rodriguez, P.: Sentiment analysis in monitoring software development processes: An exploratory case study on github's project issues. Journal of Systems and Software **104**, 82–89 (2015)
13. Kauffeld, S., Lehmann-Willenbrock, N.: Meetings matter: Effects of team meetings on team and organizational success. Small Group Research **43**(2), 130–158 (2012)
14. Kauffeld, S., Meyers, R.A.: Complaint and solution-oriented circles: Interaction patterns in work group discussions. European Journal of Work and Organizational Psychology **18**(3), 267–294 (2009)
15. Klünder, J., Prenner, N., Windmann, A.K., Stess, M., Nolting, M., Kortum, F., Handke, L., Schneider, K., Kauffeld, S.: Do you just discuss or do you solve? meeting analysis in a software project at early stages. In: Proceedings of the 5th International Workshop on Emotion Awareness in Software Engineering. ACM (2020)
16. Klünder, J., Unger-Windeler, C., Kortum, F., Schneider, K.: Team meetings and their relevance for the software development process over time. In: Proceedings of the 43rd Euromicro Conference on Software Engineering and Advanced Applications. pp. 313–320. IEEE (2017)
17. Kraut, R.E., Streeter, L.A.: Coordination in software development. Communications of the ACM **38**(3), 69–82 (1995)
18. Kuhrmann, M., Tell, P., Klünder, J., Hebig, R., Licorish, S.A., MacDonell, S.G.: Complementing materials for the helena study (stage 2). [online] DOI: 10.13140/RG.2.2.11032.65288 (November 2018)
19. Lehmann-Willenbrock, N., Meyers, R.A., Kauffeld, S., Neininger, A., Henschel, A.: Verbal interaction sequences and group mood: Exploring the role of team planning communication. Small Group Research **42**(6), 639–668 (2011)
20. 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 (2018)
21. Marjaie, S., Rathod, U.: Communication in agile software projects: Qualitative analysis using grounded theory in system dynamics. In: Proceedings of the International Conference of the System Dynamics Society (2011)
22. McChesney, I.R., Gallagher, S.: Communication and co-ordination practices in software engineering projects. Information and Software Technology **46**(7), 473 – 489 (2004)
23. Oshri, I., Kotlarsky, J., Willcocks, L.P.: Global software development: Exploring socialization and face-to-face meetings in distributed strategic projects. The Journal of Strategic Information Systems **16**(1), 25–49 (2007)

24. Prenner, N., Klünder, J., Schneider, K.: Making meeting success measurable by participants' feedback. In: Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering. ACM (2018)
25. Remus, R., Quasthoff, U., Heyer, G.: Sentiws-a publicly available german-language resource for sentiment analysis. In: LREC. Citeseer (2010)
26. Schneider, K., Klünder, J., Kortum, F., Handke, L., Straube, J., Kauffeld, S.: Positive affect through interactions in meetings: The role of proactive and supportive statements. Journal of Systems and Software **143**, 59–70 (2018)
27. Teasley, S., Covi, L., Krishnan, M.S., Olson, J.S.: How does radical collocation help a team succeed? In: Proceedings of the 2000 ACM conference on Computer supported cooperative work. pp. 339–346 (2000)
28. Waltinger, U.: Sentiment analysis reloaded-a comparative study on sentiment polarity identification combining machine learning and subjectivity features. In: WEBIST (1). pp. 203–210 (2010)
29. Wang, H., Castanon, J.A.: Sentiment expression via emoticons on social media. In: 2015 ieee international conference on big data (big data). pp. 2404–2408. IEEE (2015)
30. Watzlawick, P., Bavelas, J.B., Jackson, D.D.: Pragmatics of human communication: A study of interactional patterns, pathologies and paradoxes. WW Norton & Company (2011)
31. Zheng, J., Veinott, E., Bos, N., Olson, J.S., Olson, G.M.: Trust without touch: jumpstarting long-distance trust with initial social activities. In: Proceedings of the SIGCHI conference on human factors in computing systems. pp. 141–146 (2002)