What Is Unclear?
Computational Assessment of Task Clarity in Crowdsourcing

Zahra Nouri
Department of Computer Science, Paderborn University
Paderborn, North Rhine-Westphalia, Germany
znouri@mail.upb.de

Ujwal Gadiraju
Web Information Systems, Delft University of Technology
Delft, Netherlands
u.k.gadiraju@tudelft.nl

Gregor Engels
Department of Computer Science, Paderborn University
Paderborn, North Rhine-Westphalia, Germany
engels@upb.de

Henning Wachsmuth
Department of Computer Science, Paderborn University
Paderborn, North Rhine-Westphalia, Germany
henningw@upb.de

ABSTRACT

Designing tasks clearly to facilitate accurate task completion is a challenging endeavor for requesters on crowdsourcing platforms. Prior research shows that inexperienced requesters fail to write clear and complete task descriptions which directly leads to low quality submissions from workers. By complementing existing works that have aimed to address this challenge, in this paper we study whether clarity flaws in task descriptions can be identified automatically using natural language processing methods. We identify and synthesize seven clarity flaws in task descriptions that are grounded in relevant literature. We build both BERT-based and feature-based binary classifiers, in order to study the extent to which clarity flaws in task descriptions can be computationally assessed, and understand textual properties of descriptions that affect task clarity. Through a crowdsourced study, we collect annotations of clarity flaws in 1332 real task descriptions. Using this dataset, we evaluate several configurations of the classifiers. Our results indicate that nearly all the clarity flaws in task descriptions can be assessed reasonably by the classifiers. We found that the *content*, *style*, and *readability* of tasks descriptions are particularly important in shaping their clarity. This work has important implications on the design of tools to help requesters in improving task clarity on crowdsourcing platforms. Flaw-specific properties can provide for valuable guidance in improving task descriptions.

CCS CONCEPTS

• **Computing methodologies → Natural language processing**;
• **Human-centered computing**;

KEYWORDS

Crowdsourcing, Task design, Unclear task descriptions, Task clarity assessment, Feature-based binary classification, BERT-based binary classification

HT '21, August 30-September 2, 2021, Virtual Event, Ireland
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8551-0/21/08…$15.00
https://doi.org/10.1145/3465336.3475109

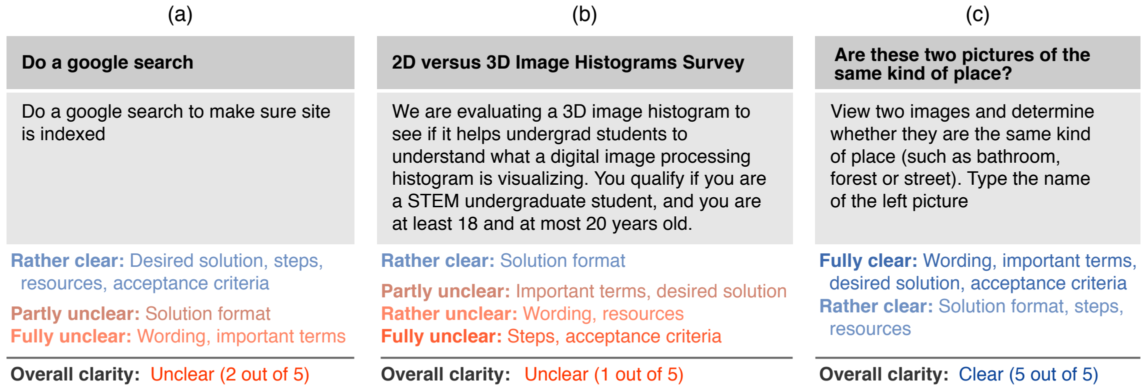
ACM Reference Format:
Zahra Nouri, Ujwal Gadiraju, Gregor Engels, and Henning Wachsmuth. 2021. What Is Unclear? Computational Assessment of Task Clarity in Crowdsourcing. In *Proceedings of the 32nd ACM Conference on Hypertext and Social Media (HT '21), August 30-September 2, 2021, Virtual Event, Ireland*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3465336.3475109

1 INTRODUCTION

Crowdsourcing is growing extensively and has been greatly beneficial to organizations and individuals alike [33]. Crowdsourcing marketplaces facilitate on-demand access to diverse human input, having led to a vast family of cost-effective solutions and services. This flourishing paradigm provides the potential to harness the wisdom, abilities, and creativity of a crowd for problems that require human intelligence. The general crowdsourcing process includes three main phases [50]: (1) *Task design*, where requesters post task descriptions on a crowdsourcing platform. (2) *Task operation*, where workers accept tasks and then submit their results. Workers may ask questions about task details, and requesters may give feedback on the results. And (3) *task evaluation*, where requesters decide to accept results and to pay workers.

The majority of prior research on crowdsorucing has focused on the quality of results provided by crowd workers [41]. Low-quality solutions are considered as the major barrier to achieve the full potential of crowdsourcing [57]. This problem emerges from challenges pertaining to the three main stakeholders involved: (1) *workers* may be unqualified, inexperienced, or unmotivated to complete tasks effectively [16, 31, 36, 52, 58]; (2) *requesters* may also be inexperienced, unfair, or sometimes careless in task design, task operation, and task evaluation [11, 36, 50, 58]; and (3) the *platform* may mediate the entire crowdsourcing process poorly, especially in terms of facilitating requester-worker communication [50]. Among several factors that have been shown to shape the quality of crowd work, unclear task design has been highlighted as one of the most critical [27, 41, 43, 48, 58]. Poor task design can lead to disappointment and frustration among workers due to a misalignment of expectations and unwarranted rejection of work [25, 40]. As argued by prior work, this can harm the relationship between requesters and workers, destabilizing the dynamics of crowd work in the long run [20, 21, 49].

The creation of clear task descriptions is therefore crucial for an effective task design. Usually, such a description consists of a



(a)

Do a google search

Do a google search to make sure site is indexed

Rather clear: Desired solution, steps, resources, acceptance criteria

Partly unclear: Solution format

Fully unclear: Wording, important terms

Overall clarity: Unclear (2 out of 5)

(b)

2D versus 3D Image Histograms Survey

We are evaluating a 3D image histogram to see if it helps undergrad students to understand what a digital image processing histogram is visualizing. You qualify if you are a STEM undergraduate student, and you are at least 18 and at most 20 years old.

Rather clear: Solution format

Partly unclear: Important terms, desired solution

Rather unclear: Wording, resources

Fully unclear: Steps, acceptance criteria

Overall clarity: Unclear (1 out of 5)

(c)

Are these two pictures of the same kind of place?

View two images and determine whether they are the same kind of place (such as bathroom, forest or street). Type the name of the left picture

Fully clear: Wording, important terms, desired solution, acceptance criteria

Rather clear: Solution format, steps, resources

Overall clarity: Clear (5 out of 5)

Figure 1: Example crowdsourcing task descriptions from the dataset introduced in Section 5. The labels below the descriptions capture how annotators assessed the different dimensions that we propose to model the clarity of task descriptions on average.

title along with a body containing instructions. In general, a task description should be easy to understand and follow, and should describe sufficiently what is expected to be done by workers and how this should be done [9, 22, 24, 29, 40, 47]. The description immediately impact workers' perception and selection of a task [46, 54], with consequences for their participation [38] and task completion rate [15], as well as for their approval rate, reputation, and income [55] despite spending time and efforts [48]. All of this affects both the expected quality of results and the workers' trust and satisfaction [22, 58]. Accordingly, Khanna et al. [38] presented evidence that *task clarity* through the descriptions improve the usability of crowdsourcing, especially when hiring low-income workers.

Consequently, achieving task clarity is of great importance to crowdsourcing processes. Yet, prior research on crowdsourcing emphasizes that ambiguous task descriptions have been a constant challenge [13, 23, 27, 28, 38, 50, 58]. On one hand, requesters are required to adequately describe all information necessary for completing a given task, including the resources to use, the steps to be followed, and the solution format to submit. This is often difficult without extensive crowdsourcing experience, especially for micro-tasks that are meant to appeal to a broad range of workers, spanning diverse cultures, possessing different skills, and varying educational backgrounds [35]. On the other hand, writing down task descriptions in a clear and understandable way is challenging, due to both the subjectivity of language used by task requesters and the inherent ambiguity of natural language in general. Thus, workers may interpret the instructions and requirements they read differently [24]. Consider the simple task description in Fig. 1(a), for example. Even if we expect that the "site" referred to is given, what solution is required to "make sure" that the site is indexed may be unclear to workers, let alone what the technical term "indexed" exactly means. The notably longer task description in Fig. 1(b) shows similar clarity flaws, and it even leaves the overall task goal fully unclear (i.e., what needs to be achieved for acceptance and how this can be achieved). In contrast, the task description in Fig. 1(c), though certainly not perfect, clearly states what is to be done and how, using easy language without any complex terms.

We argue that the dual problem of describing all necessary information in unambiguous phrasing calls for technological assistance of requesters that helps them maximize the clarity of their task descriptions while obtaining the level of completeness required. Task requesters could greatly benefit from interactive tools that automatically assist them in improving the quality of descriptions. To our knowledge, however, such tools have not been built due to a lack of usable computational methods that can assess the clarity of task descriptions.

In this paper, we aim to investigate the extent to which the clarity of a crowdsourcing task description can be assessed by leveraging natural language processing methods. Concretely, we address the following research questions:

- **RQ1.** How effectively can the most common clarity flaws in task descriptions be identified automatically?
- **RQ2.** What textual properties render a task description unclear with respect to the defined flaws?

To contribute to the state of the art in task design support (Section 2), we define a set of common task clarity flaws, covering difficult wording, missing definition of terms, specifications of the desired solution, the solution format, the steps to perform, and the required resources, as well as the criteria to meet for task acceptance (Section 3). We hypothesize all of them to negatively affect overall clarity, i.e., to lead to unclear task descriptions. For computational task clarity assessment (RQ1), we develop two natural language processing approaches (Section 4): a transformed-based neural model, BERT [18] and a linear SVM with six feature types covering both standard content and style aspects as well as clarity flaw-specific indicators. The latter particularly helps to explore the impact of textual properties in light of RQ2. Since no data for studying task clarity exists yet, we extended 1332 real micro-task descriptions from an existing dataset [19, 27] for clarity flaws in a crowd-based annotation task (Section 5). Each description was assessed for all flaws and overall unclarity by five workers on a scale from 1 to 5. Our correlation analysis suggests that all flaws can affect clarity, while none of them is decisive alone.

Given the dataset, we experiment with BERT as well as one SVM with each feature type alone, feature ablation (all but one), and all features for binarity clarity flaw and overall unclarity classification (Section 6). With respect to RQ1, BERT models perform with test set accuracy ranging from 0.55 to 0.71 while SVMs outperform both with accuracy ranging from 0.61 to 0.74 for all cases except the undefined important term flaw. Both approaches show learning success in almost all cases but difficult wording flaw which seems hard to assess, possibly due to imbalanced data distributions. With respect to RQ2, we observe that the content (reflected by TF-IDF features), style and readability seem important, but also explicitly checking for flaw-specific indicators helps in many cases. In contrast, the length of task descriptions seems to have little effect on their clarity.

Altogether, the main contributions of this paper are:[1]

- A dataset with 1332 real task descriptions, manually annotated for studying task clarity in crowdsourcing.
- A feature-based and a neural approach for computational assessment of task clarity.
- Extensive empirical insights into what helps in computational assessment of task clarity and to what extent.

2 RELATED WORK

Research on text clarity in general has investigated understandability and readability with respect to the syntax and semantics of text [39], vocabulary that causes semantic difficulties [12], the use of statistical language models for assessment [17], and more. Kevyn [37] summarized a wide range of works on analyzing the readability of text. Here, we position our work with respect to related literature that has explored task clarity in crowdsourcing marketplaces, in terms of (1) workflows, (2) methods or models, and (3) tools. We discuss task clarity further in Section 3.

2.1 Workflows

Salehi et al. [53] proposed a workflow for complicated writing tasks where workers first post their questions regarding the task to the requester. After discussing the questions with the requester, workers write a draft. Then, the requester rates the drafts and discusses them with the workers. The workers, in turn, edit the paragraphs based on the ratings and discussion and submit the final paragraph. This workflow is expensive in terms of time and money. Moreover, it highly relies on the workers' communication and relation to the requester as well as on a well-structured feedback mechanism which seems to be challenging to achieve. Similarly, Manam et al. introduced TaskMate, a workflow that depends on the workers to improve the clarity of task descriptions. TaskMate offers a collaborative setting in which workers identify the ambiguities of a task description in the form of questions and suggest several answers for each question, other workers vote for the answers that most probably clarify the ambiguities, and a clearer task description is then created to be done by the workers [47]. This approach reduces the effort for requesters to create clear instructions. It puts all the responsibility on workers, assuming that they collaborate well together. Its impact thus depends on how reliable the workers are in terms of collaboration and general work quality.

Gaikwad et al. [28] defines a workflow called Daemo which offers requesters to post several instances of their task, receive feedback from workers, on which basis they can optimize their task description. Experiments indicate that this method is effective in principle. Despite the cost in terms of time and money for this pilot step, however, the approach may not be suitable for large crowds with diverse backgrounds and skills, due to its reliance on the subjective judgements of a limited number of workers submitting their feedback.

In contrast, the computational approach we present in this paper proposes an automated tool that offers a faster and cheaper workflow on the requester's side, since it fully removes the workers from the process. The intended workflow is independent from various challenges discussed in previous work [50], ranging from low-quality results submitted by workers, complications of poor requester-worker relationship, and the lack of a proper feedback system in the process. If helpful, it may still be followed by workflows or interventions that include the workers' opinions.

2.2 Models and Methods

A number of researchers have proposed models of what task clarity means as well as methods to analyze task clarity. Among these, Gadiraju et al. [27] proposed a computational model, which stresses predictive features such as role clarity and goal clarity as the major aspects of task clarity in crowdsourcing. Additionally, the authors presented a means to quantify clarity. Wu and Quinn [58] measured the effect of guidelines on workers' perception of task quality and, consequently, on the outcome of the task including accuracy, throughput, trust, and worker satisfaction. Papoutsaki et al. [51] studied inexperienced requesters in a data collection task; how they design a task, what dimensions they consider, and similar. Their analysis contains helpful lessons for novice requesters. Khanna et al. [38] showed that the user interface, task instructions, and the workers' cultural background prevent workers with low digital abilities from accepting and doing tasks on Mturk. They recommend to simplify the user interface and task instructions, along with language localization to harness the usability of workers. Finnerty et al. [22] provided evidence that clear instructions and a simpler task design which encourages the workers' focus and awareness leads to higher quality results.

To build computational methods, we extended the clarity aspects discussed by Gadiraju et al. [27]. Accordingly, we annotated the dataset they used in their work in order to investigate the assessibility of descriptions' clarity flaws by only processing the plain text of the descriptions. We believe that the trained classifiers we present below will advance the development of assistance tools that are not only helpful for inexperienced requesters to improve the clarity of their task descriptions, but also for workers to better understand tasks and, hence, to deliver more accurate results.

2.3 Tools

Manam and Quinn [48] introduced WingIt, a system that relies on the workers' assumptions and intuitions about the task requirements as well as the requesters' desired result in the case of ambiguous task instructions. WingIt enables workers to either ask the requester for clarifications in the form of questions ("Q&A") with the best answer that clarifies the ambiguity, or directly revise the instructions ("Edit"). The worker either waits for requester's response within three minutes (synchronous) or submits the result

[1] The Data and experiment code are available here: https://osf.io/m8njv/

assuming the requester confirms the answer (asynchronous). Another approach is implemented in the SPROUT system [11] which organizes confusing questions and utilizes suggestions from crowd workers to edit ambiguous parts of task instructions. It provides requesters with the questions and allows them to prioritize the set of questions. These tools can potentially involve inexperienced workers and require additional time from workers and costs from requesters. Yet, the risks of misunderstandings and wrong perceptions of workers remain, which may consequently lead to rejection and a bad reputation.

Kulkarni et al. [44] introduced a system called Turkomatic which works based on a price-divide-solve algorithm. Via step-by-step guidance, it utilizes the crowd to decompose and solve complex tasks provided by requesters. Turkomatic requires knowledgeable workers, supervision from the requesters, and a close feedback mechanism to be successful. With a similar idea, Chang et al. [14] proposed a collaborative system called Revolt which focuses on image-labeling tasks with vague or incomplete instructions. In Revolt, multiple workers are allowed to label the task with the given steps and have access to the description written by other workers. In case of a conflict, workers relabel the image according to other workers' descriptions. Also, the system Fantasic [30] tests a task design to help novice requesters. It collects task requirements from requesters to then create and show a task description before posting on the platform, but this is limited to a narrow set of task types. Other systems include Soylent, a plug-in for Microsoft Word through which with the help of workers documents are edited, shortened, and proofread by hiding the complexity of task specification [10], as well as CrowdForge [42], and Crowd4u [34]. The two latter help to decompose complex tasks written in natural language into small tasks for crowdsourcing platforms, but they lack generalizability to all types of task specifications in crowdsourcing. TurKit, finally, helps requesters with iterative task deployment on MTurk [46]. Its architecture avoids receiving redundant submissions by saving intermediate results. TurKit makes the assumption that the way tasks are broken down will be determined by requesters in all cases.

In contrast, we develop natural language processing methods to automatically detect clarity flaws in task descriptions independent of workers' involvement and the necessity of their communication with requesters. In future, these automatic methods may be deployed to develop a tool by which requesters can identify and improve their task description's clarity weaknesses before its publication on the platform. By avoiding worker interaction in the process, it is not only more efficient in terms of time and cost, but it may also increase effectiveness, because it is not affected by the various challenges arising out of workers' and platforms' involvement along with the complications that workers face with requesters in the process [50]. This may particularly help inexperienced requesters to learn what information is necessary for creating complete and unambiguous task descriptions.

3 ASSESSMENT OF TASK CLARITY

In this section, we present *task description clarity assessment*, the task tackled in this paper. First, we briefly discuss the notion of task description clarity in crowdsourcing. How can we design textual descriptions that contain all information relevant to complete a crowdsourcing task successfully, while being easy to understand for diverse crowd workers? Next, based on a synthesis of related literature we identify flaws with respect to task clarity and characterize unclear task descriptions.

3.1 Clarity of Task Descriptions

With *task clarity* we refer to a twofold property of crowdsourcing task descriptions, which not only influences the level of comprehensibility and completeness of the instructions written in natural language, but also determines the extent to which information required for delivering a high-quality result to the task is provided by its requester. In addition to determining participation criteria and laying down eligibility constraints for crowdsourcing tasks (in terms of reputation, experience, demographic variables, language proficiency, and similar), task clarity is primarily shaped by the *task design* of requesters in crowdsourcing marketplaces.

In particular, problems with task clarity may occur due to the inexperience of task requesters, who may lack an understanding of the diversity among target workers in terms of their background knowledge, skills, demographics, and culture. Similarly, they may lack sufficient awareness of the importance of a thorough task design and its direct influence on the quality of the submissions from workers. Unclear task descriptions can lead to inaccurate or incorrect responses from workers, which can in turn lead to task rejection and distrust between requesters and workers.

Different researchers have addressed challenges pertaining to task clarity, and studied dimensions that lead to instructions being perceived as vague or incomplete. Among these, Gadiraju et al. [27] discussed *goal clarity* and *role clarity* as main dimensions of microtask descriptions. These terms refer to what is expected to be delivered from workers and how the work is planned to be done, respectively. Moreover, Wu and Quinn [58] introduced the notion of *descriptive metrics* and *prospective metrics* of task descriptions. For identifying the task clarity flaws, descriptive metrics are of particular interest. They include (a) the vocabulary or language used to describe the task, (b) the specification of the data that is expected to be delivered by crowd workers, and (c) the order of the steps that should be carried out in a task and the solution to be submitted. Prospective metrics, on the other hand, refer to the task properties which are more subjective relating to the workers' personal feelings. Such metrics play a role when it comes to workers' trust, confidence, and prediction of outcomes rather than their comprehension of the task, influenced directly by task descriptions in general. [2]

Finally, other information adhering to best practices for a complete task instruction includes the interface on which the work should be performed, the expected format of the solution, and the specification of acceptance conditions [58].

3.2 Clarity Flaws: Characterizing Unclear Tasks

Based on the prior studies (Section 3.1) on the characteristics of incomplete and unclear tasks, we formed a set of clarity flaws that

[2] Occasional exceptions may occur for tasks where the concrete object of investigation is specifically known to a worker. However, we are interested in the general form of a description rather than its concrete object of investigation.

define the basis for annotation guidelines in our dataset creation in Section 5, We here propose to model clarity by assessing the following clarity flaws. The first one, *description unclear*, can be understood as an overall unclarity assessment, while the remaining reflect sub-dimensions of unclear task descriptions:

(1) **Description unclear.** The task is unclear, i.e., it is not fully understandable how one can complete the task successfully and/or what the desired solution is. This refers to overall unclarity.
(2) **Difficult wording.** The words and the grammatical constructions used in the task descriptions are not fully comprehensible.
(3) **Important terms undefined.** Some terms that are potentially important to properly understand the tasks are not defined sufficiently. This refers to the vocabulary used for the description [29].
(4) **Desired solution unspecified.** The solution that is actually desired to be submitted in response to a task is not explained in sufficient detail. This refers to the goal clarity property.
(5) **Solution format unspecified.** The format in which the solution should be submitted is not specified sufficiently. This refers to the required detailed information regarding the goal clarity.
(6) **Steps unspecified.** The steps that need to be carried out one after another to complete a task are not defined sufficiently. This refers to the role clarity property.
(7) **Resources unspecified.** The resources that are required to be used to complete the task are not sufficiently specified. Resources may include data, tools, links, websites, etc. This refers to the necessary data and links to perform the task.
(8) **Acceptance criteria unspecified.** The acceptance criteria on which basis a requester decides about the acceptance of a solution submitted to a task are not sufficiently specified. This refers to the information which can help decide how much time to spend for performing the task and how much the work is rewarded [43].

4 COMPUTATIONAL APPROACHES TO THE ASSESSMENT OF TASK CLARITY

The main goals of this paper are to study how well computational methods can assess the aforementioned clarity flaws based only on the plain text task descriptions, and what textual properties of descriptions indicate the task clarity flaws. We investigate two approaches for our objectives, a state-of-the-art neural model and a traditional feature-based model. Both are motivated and detailed in the following section.

4.1 Neural and Feature-based Clarity Assessment: Estimating Possible Effectiveness

In this paper, we employ two approaches to compare their effectiveness for clarity assessment in order to address Research Question RQ1 from Section 1: (a) we rely on transformer-based neural models which have been shown to be superior in a variety of natural language processing tasks, (b) we use linear SVM classifiers based on six feature types, since they are found to be effective if data is limited.

For neural approach, we rely on the widely used BERT model [18]. We explore two common variations of pre-trained BERT namely *Bert-base-uncased*, a case-insensitive model trained on lower-cased English text, and *Bert-base-cased*, a case-sensitive model trained on English text in its original format. Both variations have 12 layers, 768 hidden nodes, 12 heads, and almost 110 million parameters [3]. For feature-based approach, we collected six specific feature types based on which the classifiers assess the task clarity. In the following, we introduce the feature types in details.

4.2 Feature-based Clarity Assessment: Studying Textual Properties

Feature-based classification can provide detailed insights into the textual properties that are helpful in assessing each clarity flaw, allowing us to address RQ2. Concretely, our linear SVM classifiers are based on the following six feature types. These types cover both, standard features that have often been used in natural language processing and clarity flaw-specific features that we engineered based on the well-known aspects of task descriptions [27]:

(1) **Content.** Content is important in many text classification tasks. Consistent with common practice, we examine the effect of content-related properties via term frequency–inverse document frequency (TF-IDF) where we consider all lower-cased token 1- to 3-grams including stop words as terms.
(2) **Length.** To test whether clarity correlates with length, we include 26 features that reflect the extent of a task description. They cover the numbers of all characters, letters, digits, punctuation marks, whitespaces, non-whitespaces, unique words, words, fully upper-cased tokens, fully lower-cased tokens, capitalized words, phrases, and sentences. Additionally, we computed the mean of all counts per sentence (except for sentences).
(3) **Style.** Clarity may be considered a property of style. We hence model style, namely via part-of-speech 1- to 3-grams and phrase 1- to 3-grams (created using the NLTK library [1]) as well as characters 3-grams and the *functional words*. For the latter, we consider the top-100 most frequent lower-cased words in the whole corpus.
(4) **Subjectivity.** Subjective phrasing of task instructions have been shown to affect perceived task clarity [27]. We capture subjectivity using the *Textblob* library [8] which computes a subjectivity score, a polarity score, a negativity score, a positivity score, and an objectivity score for a given text.
(5) **Readability.** As mentioned in Section 2, readability metrics have been used for clarity assessment. We consider Flesch-Kincaid Grade Level, ARI, Coleman-Liau, Flesch Reading-Ease, Gunning-Fog Index, LIX, SMOG Index, RIX, and Dale-Chall Index. All readability metrics are computed via the Pypi library [4].
(6) **Flaw-specific.** In line with the clarity flaws from Section 3, we hypothesize that the clarity of task descriptions is reflected in the *completeness*, in terms of resources and acceptance criteria, as well as the *complexity* of words and terms.

Table 1: The distribution of crowdsourcing task descriptions over the six different task types in the original dataset [27], after filtering out near-copies, and in the final 50% sample that we used for our annotated dataset.

#	Task Type	# Original	# No Near-Copies	# Our Dataset
SU	Surveys	1200	1121	561
CA	Content Access	1008	528	264
IA	Interpretation and Analysis	1199	505	253
IF	Information Finding	1200	291	144
CC	Content Creation	1200	147	74
VV	Verification and Validation	1200	71	36
	Total	**7007**	**2663**	**1332**

To study completeness and complexity, we introduce the following eight task-specific features. The first four are binary features capturing whether a description matches the given regular expression.

a. *Website.* Regular expression for various token 1- or 2-gram which may refer to a web resource (e.g., "web page(s)", "webpage(s)", "site(s)", "web site(s)", etc.).
b. *Link.* Regular expression for URls and placeholder words such as "link".
c. *Given time.* Regular expression for token 1-grams delivering information regarding the estimated time to complete, such as "5 minutes", "1 minute 14 seconds", "2 min".
d. *Reward.* Regular expression for token 1-grams delivering information regarding the specified reward (or bonus) for a task, such as "up to $0.57 + 50% bonus = $0.85 max", "5 cents", "avg rwrd+bns: $2.02"
e. *Entity.* All token *n*-grams detected by Spacy [6] as locations, organizations, ordinal entities, products, or similar.
f. *POS categories.* Frequencies of conceptually similar part-of-speech tags found with by the Stanford Tagger [7], such as verbs, nouns, open and close part-of-speech tags, and similar.
g. *Discrete words.* The 10 most frequent discrete lower-cased 1-gram tokens (excluding stop words) which appear either only in clear task descriptions or only in unclear descriptions in all dimensions.
h. *Complex words.* Two different scores for the complexity of token 1-grams computed by Pypi [4] .

5 A DATASET FOR STUDYING THE ASSESSMENT OF TASK CLARITY

To allow studying the clarity assessment of task descriptions, we created and validated a new dataset in four main steps: (1) the compilation of task descriptions, (2) the annotation of the descriptions for clarity flaws, (3) the consolidation of the final dataset, and (4) a basic correlation analysis of the clarity flaws. In the following, we detail each step, describing the source data and annotation process as well as the resulting data distribution and correlations.

5.1 Compilation of Crowdsourcing Task Descriptions

For our data compilation, we built on the previously published dataset of Gadiraju et al. [27] and Difallah et al. [19], which consists of a total of 7007 records of real task descriptions published on Amazon Mechanical Turk (mTurk) from October 2013 to September 2014.[3] For each task, the title, body, date of publication, and some other metadata are given. For our study, the title text, a dot (as separator), and the body of the tasks compose the task descriptions in the dataset. The task descriptions are grouped into six different task types, namely *Surveys (SU)*, *Content Access (CA)*, *Interpretation and Analysis (IA)*, *Information Finding (IF)*, *Content Creation (CC)*, as well as *Verification and Validation (VV)* [26].

During inspection of the original dataset, we observed that the 7007 task descriptions contains a lot of cases that are near-copies of others in terms of being multiple instances of the same task only with specific information replaced. Since we did not expect any clarity-specific differences in these, we filtered out all near-copies in a semi-automatic process, ending up in 2663 clearly distinct task descriptions. Due to the limited budget, we decided to select a 50% sample for manual annotation (i.e., 1332 records). To preserve the full diversity of task descriptions covered, we chose the sample representative with respect to the task types in the filtered set. Table 1 shows the distribution of the task types in our dataset in comparison to the source data. The 1332 task descriptions span 31,027 tokens (23.3 tokens per description on average), and cover 25,891 unique tokens.

5.2 Crowd-based Annotation for Clarity Flaws

We decided to collect annotations for clarity flaws in the task descriptions directly from crowd workers, since they are eventually the ones to benefit from improved task descriptions, making their judgment decisive. In accordance with the source of the given task descriptions, we deployed the annotation tasks on Mturk, so that our annotators match potential workers for the described tasks in principle.

Task Design. Being task requesters ourselves in this setting, we took care to avoid the clarity flaws identified in Section 3 in our

[3] We are aware that the age of the data may impact what we observe. However, we decided to favor comparability to previous work over timeliness, also because we do not see a fundamental change in task descriptions after 2014. Besides, note that obtaining task descriptions is all but straightforward.

Table 2: (a) Distribution of the MACE aggregate Likert scores [32] over all 1332 task descriptions in our dataset for each clarity flaw (the higher the score, the more the flaw was observed by the annotators). (b) The corresponding binary scores, where 1 and 2 are mapped to *Negative* for without flaws, and 3, 4, and 5 to *Positive* for with flaws. The majority values are marked in bold.

		(a) 5-point Likert Scores					**(b) Binary classes**	
#	**Clarity Flaws**	**1**	**2**	**3**	**4**	**5**	**Negative**	**Positive**
1	Descriptions unclear (overall unclarity)	**0.48**	0.18	0.09	0.20	0.05	**0.66**	0.34
2	Difficult wording	**0.42**	0.26	0.05	0.03	0.24	**0.68**	0.32
3	Important terms not defined	**0.31**	0.28	0.07	0.10	0.24	**0.59**	0.41
4	Desired solutions not specified	0.24	**0.29**	0.16	0.22	0.09	**0.53**	0.47
5	Solution format not specified	**0.32**	0.27	0.18	0.07	0.16	**0.59**	0.41
6	Steps not specified	0.15	**0.35**	0.24	0.09	0.16	**0.50**	**0.50**
7	Resources not specified	0.18	**0.32**	0.25	0.13	0.12	**0.50**	**0.50**
8	Acceptance criteria not specified	**0.27**	0.21	**0.27**	0.13	0.12	0.48	**0.52**

task design. In our annotation task, the workers had to assess a given task description from our dataset for the clarity flaws via a survey-like form. In the beginning, general instructions were given on how to fill the form (along with a privacy guarantee), asking the workers to put themselves into the role of the worker who takes on the task corresponding to the description. Then, each flaw was described following the definitions from Section 3.

To determine a suitable setting in terms of the annotation scheme to use and the number of annotators to employ, we designed and deployed the annotation tasks in two phases: first, a *pilot annotation study* where we explored initial design decisions on Mturk to test whether our tasks lead to the desired results with sufficient quality; and second, the *main annotation study* where we collected the annotations of all 1332 task descriptions after improving the task design based on the findings from the pilot study.

Pilot Annotation Study. We compared two different annotation schemes in terms of which one leads to a higher inter-annotator agreement: (a) *binary classification* where workers either agreed or disagreed with statements covering each clarity flaw; and (b) *5-point Likert scoring* where workers reported the extent to which they agreed with the statements from “1: strongly disagree” to “5: strongly agree”. We then carried out a pilot study with two batches of 12 annotation tasks (one for binary, one for Likert). Each annotation task included four task descriptions, meaning 48 descriptions in total. In addition to the flaw assessments, the workers were asked to give a summary of the task description, which we used as a quality check to see whether they carefully read the task descriptions while deciding on the labels. We estimated 8 minutes to complete each task and paid USD 1.32 per task to each worker. Each task was to be annotated by three workers with more than 1000 approved tasks (HITs) on mTurk, as suggested by mTurk to ensure the quality of annotators based on their reputation.

Our analysis of the results of the pilot study revealed that the 5-point Likert scale gave workers more freedom to make more accurate judgement, while leading to higher inter-annotator agreement; for the annotated flaws, we observed full agreement ranging from 40% to 63% for the binary scheme, and from 50% to 75% for 5-point likert scoring after discarding unreliable workers. Moreover, the summaries written by the workers illustrated the necessity to select workers more restrictively to increase reliability.

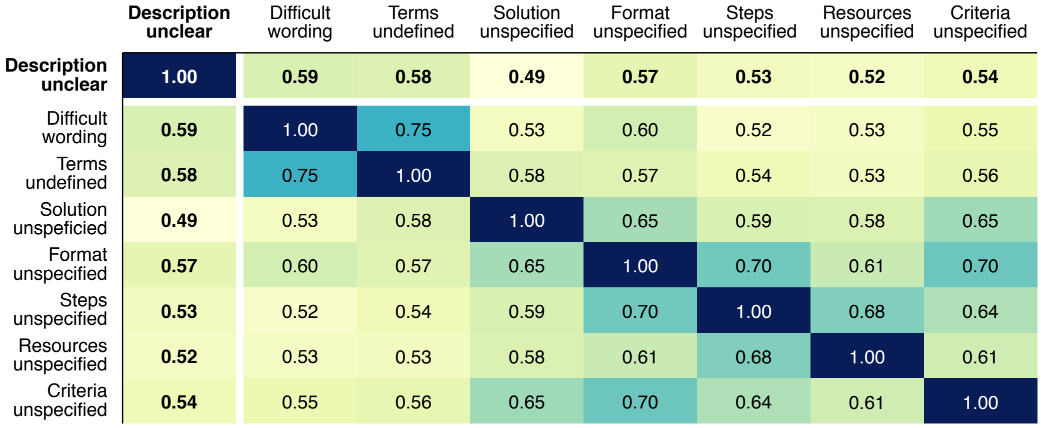
Main Annotation Study. We decided to acquire only so called *master workers* with an approval rate higher than 95% who are English speakers from the US, Canada, England, Ireland, Australia, New Zealand, or India.[4] Each task was tackled by five annotators, spending about 5 minutes on average with an hourly wage of around 10 USD per task. We refined the quality check by replacing the summary text field with two text fields for *other problems* that the workers potentially find with the description (an optional field) and *a brief suggestion* for improving the task descriptions (a mandatory field).[5] Finally, we deployed all 1332 task descriptions in 333 annotation tasks, each of which included four task descriptions to be annotated. Although, there was no limits on the number of tasks an individual worker could complete, the annotation tasks were done after 10 days and 33 unique workers participated in the study.

5.3 Consolidation of the Dataset

The distribution of the collected annotations for the clarity flaw *Description unclear* turned out to be unexpectedly skewed toward “strongly disagree”. We found three workers who had participated in annotating more than 150 task descriptions and had selected strongly disagree for all the statements of more than 95% of the task descriptions. To improve data quality, we discarded all assignments of these workers, still remaining with at least three annotations for all task descriptions. To obtain a single final annotation from the remaining annotations of each tasks, we relied on the *multi-annotator competence estimation (MACE)* [32]. MACE was designed particularly for crowdsourcing settings, where standard inter-annotator reliability measures such as Fleiss' κ and Krippendorff's α do not apply due to varying annotator sets. It grades the reliability of workers based on their agreement with others and allows deriving

[4] The title *master worker* is given to workers by a blackbox algorithms of the mTurk platform based on the workers' performance.

[5] For the study at hand, we used these texts only evaluate the reliability of workers, but an analysis of the problems and suggestions given by the workers may be interesting in future work.



	Description unclear	Difficult wording	Terms undefined	Solution unspecified	Format unspecified	Steps unspecified	Resources unspecified	Criteria unspecified
Description unclear	**1.00**	**0.59**	**0.58**	**0.49**	**0.57**	**0.53**	**0.52**	**0.54**
Difficult wording	**0.59**	1.00	0.75	0.53	0.60	0.52	0.53	0.55
Terms undefined	**0.58**	0.75	1.00	0.58	0.57	0.54	0.53	0.56
Solution unspeficied	**0.49**	0.53	0.58	1.00	0.65	0.59	0.58	0.65
Format unspecified	**0.57**	0.60	0.57	0.65	1.00	0.70	0.61	0.70
Steps unspecified	**0.53**	0.52	0.54	0.59	0.70	1.00	0.68	0.64
Resources unspecified	**0.52**	0.53	0.53	0.58	0.61	0.68	1.00	0.61
Criteria unspecified	**0.54**	0.55	0.56	0.65	0.70	0.64	0.61	1.00

Figure 2: Pearson's *r* correlation coefficient for each pair of the annotated clarity flaws in our dataset. The medium correlations suggest that all specific flaws add to some extent to overall unclarity (*Description unclear*), but none of them decides it alone.

one aggregate annotation for each instance on this basis.[6] The competence value of the annotators ranged from 0.01 to 0.97. While the average was only 0.13, the top five all had a confidence above 0.32. In addition to the resulting MACE Likert scores from 1 to 5, we also established binary class for each task description, where we consider "strongly agree" (5) and "rather agree" (4) as *Positive*, and "partly agree/partly disagree" (3), and lower as *Negative*.

Table 2 shows the distribution of scores in the consolidated dataset for each clarity flaw from Section 3. We observe that the distribution is slightly skewed towards lower scores in general, but that the whole scale is covered reasonably in most cases. Particularly, the binary classes are reasonably balanced. For development and evaluation, we split the dataset based on the tasks' publication date on mTurk, in order to simulate the idea of unseen future tasks in testing. The training set contains 666 task descriptions (50%) with 15,128 tokens, the validation set contains 333 descriptions (25%) with 7,821 tokens, and the test set also contains 333 descriptions (25%) here with 8,078 tokens.

5.4 Correlation Analysis

Given the final Likert scores, we carried out a correlation analysis for all clarity flaws, in order to roughly assess whether they can be distinguished and how well they predict that a description is unclear (overall unclarity). Figure 2 shows the Pearson's *r* correlation coefficient for each pair of flaws. Only few flaws correlate strongly with each other, the highest coefficient being observed for *Difficult wording* and *Important terms undefined* (0.75), which makes sense. Also, it seems intuitive that the flaw *Solution format unspecified* often goes hand in hand with *Steps unspecified* and *Acceptance criteria unspecified* (both 0.70). The majority of correlations is rather medium, roughly between 0.5 and 0.6. An important observation is that none of the seven specific clarity flaws is highly correlated with *Description unclear*, suggesting that all of them may somewhat add to overall unclarity. An uneasy wording seems to have a rather high impact (0.59), whereas a missing specification of the desired solution seems a little less important (0.49)—as far as single correlation coefficients allows such an inference.

6 EXPERIMENTS ON THE COMPUTATIONAL ASSESSMENT OF TASK CLARITY

We now present the empirical experiments that we carried out on the dataset from Section 5 with our approach from Section 4 in order to study the extent to which the clarity flaws in task descriptions from Section 3 can be assessed computationally. Concretely, we compare BERT models and feature-based SVMs in the binary classification of clarity flaws and overall unclarity. We investigate what degree of effectiveness can be expected for the computational assessment of task clarity in light of Research Question RQ1 from Section 1, and we explore what textual properties of descriptions are useful to identify task unclarity computationally (RQ2).

6.1 Experimental Setup

We evaluated the following configurations of the given approaches and two baselines in our experiments:

Transformer-based Models. We study how effectively the task clarity flaws can be assessed using eight BERT-based binary classifiers, one for overall unclarity and seven for the other clarity flaws. In each case, we used the pre-trained Bert-base-cased and Bert-base-uncased models, and we employed the *PyTorch* library [5] to conduct the experiments on the BERT-based classification. We first preprocessed the texts of the task descriptions using *BertTokenizer* for both pre-trained models. Then, we converted the preprocessed texts to the required data type for each model, respectively. We adapted *BertForSequenceClassification* to the binary-label setup and used the *BertAdam* optimizer to fine-tune the parameters with learning rate of 2^{-5} and a warmup of 0.1. Using the training set, we tuned the models for four epochs and computed the test set accuracy for the Bert-base-cased and Bert-base-uncased classifiers separately.

[6] For comparison, we also explored the use of majority voting instead, using the rounded mean Likert scores, in case no majority exists. However, we decided for MACE, since it not only accounts for annotator reliability, but also the distribution of scores turned out to be notably more balanced.

Table 3: Accuracy of each feature type, feature ablation, all features, and the two BERT variants in comparison to the majority baseline and the minority "baseline" for overall unclarity (*Description unclear*) and for the seven clarity flaws. The best values in each column are marked bold; the best feature and feature ablation are underlined. For *all features*, *Bert-base-cased*, and *Bert-base-uncased* significant improvements over the majority baseline are marked with ** ($p < .05$) and * ($p < .01$).

#	Approach	Description unclear	Difficult wording	Terms undef.	Solution unspec.	Format unspec.	Steps unspec.	Resources unspec.	Criteria unspec.
A_1	Content	0.72	0.71	0.66	0.60	0.56	0.59	**0.63**	0.58
A_2	Length	0.62	0.63	0.54	0.57	0.51	0.55	0.54	0.56
A_3	Style	0.66	0.67	0.61	0.61	0.60	0.57	0.59	**0.63**
A_4	Subjectivity	0.71	0.51	0.63	0.50	0.49	0.52	0.53	0.55
A_5	Readability	0.69	0.70	0.65	**0.63**	0.57	0.55	0.62	0.58
A_6	Flaw-specific	0.69	0.71	0.64	0.55	0.59	0.57	0.61	0.60
$A_{\setminus 1}$	w/o Content	0.72	0.72	0.67	0.62	0.60	0.58	0.60	0.61
$A_{\setminus 2}$	w/o Length	**0.74**	0.72	0.66	0.59	**0.62**	**0.62**	**0.63**	0.58
$A_{\setminus 3}$	w/o Style	0.69	0.72	0.67	0.61	0.59	0.59	0.61	0.61
$A_{\setminus 4}$	w/o Subjectivity	0.73	0.72	0.67	0.61	**0.62**	0.58	0.61	0.62
$A_{\setminus 5}$	w/o Readability	**0.74**	0.70	0.67	0.60	**0.62**	0.60	0.62	0.60
$A_{\setminus 6}$	w/o Flaw-specific	0.72	0.70	0.67	**0.63**	0.58	0.57	0.56	0.60
A_{1-6}	**All features**	0.73	0.74	*0.66	*0.61	*0.59	*0.61	*0.61	*0.62
BbC	**Bert-based-cased**	0.69	0.71	***0.69**	*0.60	*0.60	*0.56	*0.56	*0.57
BbU	**Bert-based-uncased**	0.71	0.71	*0.67	*0.62	*0.61	*0.58	*0.60	*0.55
Ma	Majority baseline	0.72	**0.75**	0.31	0.38	0.36	0.43	0.41	0.44
Mi	Minority "baseline"	0.28	0.25	0.69	0.62	0.64	0.57	0.59	0.56

Feature-based Models. Specifically to examine what textual properties of task descriptions are helpful for the computational assessment of task clarity, we used binary SVMs with the six feature types from Section 4.2. We used the *scikit-learn* library [2] to conduct the experiment on feature-based classification. To account for the imbalances in the data distribution, we randomly resampled the training dataset for each classifier independently. Then, we trained eight linear SVM classifiers, again one for overall clarity assessment and seven for the other flaws. We did this once for each of the six feature types alone, for each feature ablation (all but one feature type), and for all features together. We optimized the cost hyperparameter of each classifier on the validation set (tested range: 2^i for $-10 < i < 10$). Finally, we computed the test set accuracy score of each best validation set configuration on the test set.

Baselines. We compare the approaches simply to a *majority baseline*, which always predicts the majority training class. Thereby, we make visible where actual learning success is achieved. For some test sets, the majority class is different from training. To make visible where we can definitely distinguish classes, we also show the *"minority" baseline* (predicting the minority training class) below, but we point out that this is not a reasonable baseline in practice.

Significance Tests. We performed a one-tailed independent t-test to study whether (a) Bert-base-cased, (b) Bert-base-uncased, and (c) the SVM with all features can assess the task clarity flaws significantly better than the majority baseline at $p < .05$ (marked **) and $p < .01$ (*).

6.2 Results on the Effectiveness of the Assessment of Task Clarity (RQ1)

Table 3 shows all test set accuracy results for overall unclarity (*Description unclear*) and the seven clarity flaws in task descriptions, both for the BERT models and for the SVM classifiers with single feature types (A_i), feature ablation ($A_{\setminus i}$), and all features (A_{1-6}). We first analyze the overall results in light of the first research question, RQ1.

We evaluate the test set accuracy of the BERT models against the majority baseline to find out whether the considered task descriptions' clarity flaws seem possible to be identified computationally. We see that *Bert-base-cased* and *Bert-base-uncased* succeed in assessing six flaws: unspecified important terms (0.69/0.67 vs. 0.31), desired solutions (0.60/0.62 vs. 0.38), solution format (0.60/0.61 vs. 0.36), steps to perform (0.56/0.58 vs. 0.43), resources (0.56/0.60 vs. 0.41), and acceptance criteria (0.57/0.55 vs. 0.44), while they fail to improve over the majority baselines overall unclarity (0.69/0.71 vs. 0.72) and difficult wording (0.71/0.71 vs. 0.75). Moreover, the hypothetical approach of predicting the minority class would lead competitive results for some flaws. Comparing the two BERT variants, we observe that the case-insensitive variant (Bert-base-uncased) performs slightly better, showing higher results in five cases. This suggests that our choice to lower-case all words for the features was appropriate.

Given that the training data is not huge, the feature-based classifiers apparently benefit from their focused analysis; they reach higher test set accuracy compared to both BERT models for several cases: unspecified desired solution (A_5 and $A_{\setminus 6}$ with 0.63 vs. Bert-base-uncased with0.62), solution format ($A_{\setminus 2}$, $A_{\setminus 4}$, and $A_{\setminus 5}$ 0.62 vs. 0.61), steps ($A_{\setminus 2}$ 0.62 vs. 0.58), resources (A_1 and $A_{\setminus 2}$ 0.63 vs. 0.60), and acceptance criteria (A_3 0.63 vs. 0.57). For overall unclarity, the SVMs $A_{\setminus 2}$ and $A_{\setminus 5}$ perform best with 0.74 and show learning success over the majority baseline in classifying the task descriptions. Particularly the classifier without the length feature ($A_{\setminus 2}$) seems strong in general (more on the features below).

Finally, the results suggest that the clarity of having a *difficult wording* seems hard to assess; none of our approaches managed to beat the majority baseline (0.75). A reason may lie in the diversity of potentially difficulty words, which makes it hard to learn such words. However, the majority baseline result also shows that this clarify flaw shows a rather high distribution imbalance.

6.3 Results on the Impact of the Textual Properties of Task Descriptions on Task Clarity (RQ2)

The individual feature type results in the upper part of Table 3 (A_i) suggest that many of the considered textual properties are relevant to at least some of the clarity flaws. The *content* of task descriptions (measured in the form of TF-IDF) seems particularly important, achieving notably higher results than the other feature types for a number of clarity flaws, including for overall unclarity (0.72). The *style* of the descriptions performs best on *unspecified acceptance criteria* (0.63), possibly because of specific part-of-speech tags, and it is also an important indicator for an *unspecified solution format in descriptions* (0.60). Likewise, the *readability* of task descriptions helps best to identify *unspecified desired solutions* (0.63), whereas the *flaw-specific* features and the *subjectivity* of descriptions play an important role mostly in an ablation setting in the center part of Table 3 ($A_{\setminus i}$).

The insightful exception among the feature types is the *Length* (in terms of number of words, characters, digits), which achieves comparably low accuracy for all consider clarity flaws of task descriptions. This is also underlined by the feature ablation, showing that the best results overall are achieved when the length feature is left out. This finding suggests that the clarity of crowdsourcing task descriptions does not depend on their length—which is in contrast to related assessment tasks, such as predicting Wikipedia article quality [45] or argument quality [56].

7 CONCLUSIONS AND FUTURE WORK

The creation of a clear task description in terms of completeness and comprehensibility is a challenging responsibility in crowdsourcing, particularly for inexperienced requesters. Unclear or flawed task instructions can negatively impact the quality of the workers' submissions, which in turns hampers the workers' rewards and their reputation. We argue that natural language processing techniques can rise to this challenge and aid in identifying clarity flaws in task descriptions.

This paper aims to study the extent to which common clarity flaws in task descriptions can be identified automatically by leveraging natural language processing methods (RQ1), and we have investigated the textual properties of task descriptions which indicate task clarity flaws (RQ2). To this end, we have identified seven clarity flaws from relevant literature, all of which affect a task's overall unclarity (i.e., the extent to which a task description is unclear). Due to the lack of availability of a useful dataset for studying task clarity assessment, we have extended an existing dataset with flaw annotations on this basis. To this end, we have recruited crowdworkers to annotate the defined clarity flaws in 1332 real task descriptions.

Given the dataset, we have addressed RQ1 by evaluating the effectiveness of two types of computational approaches for clarity flaw assessment: transformer-based models (using BERT) as well as linear SVMs with feature types including standard content and style features as well as flaw-specific aspects. For RQ2, we carried out an individual features analysis using the SVMs, providing insights into the impact of the textual properties of descriptions on the assessment of clarity flaws. Regarding RQ1, we found that the accuracy of the BERT models ranges from 0.55 to 0.71. The SVMs outperformed the BERT, with results between 0.61 to 0.74 for most clarity flaws. Both approaches show learning success in almost all cases, except for identifying a difficult wording. Regarding RQ2, we observed that the content, style, and readability of descriptions seem to be particularly important textual properties for clarity. Combinations of the task flaw-specific properties with others are also advantageous for clarity assessment. In contrast, the length of descriptions was not found to be helpful in identifying the clarity flaws.

In the imminent future, we plan to deploy the developed methods in a tool which can help requesters to increase the clarity of their task descriptions before deploying tasks on a given crowdsourcing platform. This may eventually lead to better quality results from workers and consequently their higher reputation and satisfaction.

REFERENCES

[1] [n.d.]. Natural Language Toolkit official website. https://www.nltk.org. Accessed: 2021-04-30.

[2] [n.d.]. An open source tool for predictive data analysis. https://scikit-learn.org/stable/. Accessed: 2021-05-14.

[3] [n.d.]. Pretrained Bert models: A list of pretrained BERT models with a short presentation. https://huggingface.co/transformers/pretrained_models.html. Accessed: 2021-05-11.

[4] [n.d.]. Pypi official website, Readability project description. https://pypi.org/project/readability/. Accessed: 2021-04-30.

[5] [n.d.]. PyTorch Pretrained BERT: The Big and Extending Repository of pretrained Transformers. https://pypi.org/project/pytorch-pretrained-bert/. Accessed: 2021-05-14.

[6] [n.d.]. SpaCy official website, Named Entity Recognition documentation. https://spacy.io/models/en#en_core_web_lg. Accessed: 2021-04-30.

[7] [n.d.]. The Stanford Natural Language Processing Group official website, Stanford Log-linear Part-Of-Speech Tagger. https://nlp.stanford.edu/software/tagger.shtml. Accessed: 2021-04-30.

[8] [n.d.]. TextBlob official website: Simplified Text Processing. https://textblob.readthedocs.io/en/dev/. Accessed: 2021-04-30.

[9] Omar Alonso and Ricardo Baeza-Yates. 2011. Design and implementation of relevance assessments using crowdsourcing. In *European Conference on Information Retrieval*. Springer, 153–164.

[10] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. 2010. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 313–322.

[11] Jonathan Bragg, Daniel S Weld, et al. 2018. Sprout: Crowd-powered task design for crowdsourcing. In *The 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, 165–176.

[12] Jeanne Sternlicht Chall and Edgar Dale. 1995. *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.

[13] Jesse Chandler, Gabriele Paolacci, and Pam Mueller. 2013. *Risks and Rewards of Crowdsourcing Marketplaces*. Springer New York, New York, NY, 377–392. https://doi.org/10.1007/978-1-4614-8806-4_30

[14] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2334–2346.

[15] Jenny J Chen, Natala J Menezes, Adam D Bradley, and T North. 2011. Opportunities for crowdsourcing research on amazon mechanical turk. *Interfaces* 5, 3 (2011), 1.

[16] Chun-Wei Chiang, Anna Kasunic, and Saiph Savage. 2018. Crowd coach: Peer coaching for crowd workers' skill growth. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–17.

[17] Kevyn Collins-Thompson and James P Callan. 2004. A language modeling approach to predicting reading difficulty. In *Proceedings of the human language technology conference of the North American chapter of the association for computational linguistics: HLT-NAACL 2004*. 193–200.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[19] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. 2015. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *Proceedings of the 24th international conference on world wide web*. 238–247.

[20] Tom Edixhoven, Sihang Qiu, Lucie Kuiper, Olivier Dikken, Gwennan Smitskamp, and Ujwal Gadiraju. 2021. Improving Reactions to Rejection in Crowdsourcing Through Self-Reflection. In *13th ACM Web Science Conference 2021*. 74–83.

[21] Shaoyang Fan, Ujwal Gadiraju, Alessandro Checco, and Gianluca Demartini. 2020. CrowdCO-OP: Sharing Risks and Rewards in Crowdsourcing. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW2 (2020), 1–24.

[22] Ailbhe Finnerty, Pavel Kucherbaev, Stefano Tranquillini, and Gregorio Convertino. 2013. Keep it simple: Reward and task design in crowdsourcing. In *Proceedings of the Biannual Conference of the Italian Chapter of SIGCHI*. 1–4.

[23] Floyd Jackson Fowler Jr. 1992. How unclear terms affect survey data. *Public Opinion Quarterly* 56, 2 (01 1992), 218–231. https://doi.org/10.1086/269312 arXiv:https://academic.oup.com/poq/article-pdf/56/2/218/5222230/56-2-218.pdf

[24] Michael J Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. 2011. CrowdDB: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 61–72.

[25] Ujwal Gadiraju and Gianluca Demartini. 2019. Understanding worker moods and reactions to rejection in crowdsourcing. In *Proceedings of the 30th ACM Conference on Hypertext and Social Media*. 211–220.

[26] Ujwal Gadiraju, Ricardo Kawase, and Stefan Dietze. 2014. A taxonomy of microtasks on the web. In *Proceedings of the 25th ACM conference on Hypertext and social media*. 218–223.

[27] Ujwal Gadiraju, Jie Yang, and Alessandro Bozzon. 2017. Clarity is a worthwhile quality: On the role of task clarity in microtask crowdsourcing. In *Proceedings of the 28th ACM Conference on Hypertext and Social Media*. ACM, 5–14.

[28] Snehalkumar (Neil) S Gaikwad, Mark E Whiting, Dilrukshi Gamage, Catherine A Mullings, Dinesh Majeti, Shirish Goyal, Aaron Gilbee, Nalin Chhibber, Adam Ginzberg, Angela Richmond-Fuller, et al. 2017. The daemo crowdsourcing marketplace. In *Companion of the 2017 ACM conference on computer supported cooperative work and social computing*. 1–4.

[29] Catherine Grady and Matthew Lease. 2010. Crowdsourcing document relevance assessment with mechanical turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's mechanical turk*. 172–179.

[30] Philipp Gutheim and Björn Hartmann. 2012. Fantasktic: Improving quality of results for novice crowdsourcing users. *EECS Dept., Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2012* 112 (2012).

[31] Lei Han, Eddy Maddalena, Alessandro Checco, Cristina Sarasua, Ujwal Gadiraju, Kevin Roitero, and Gianluca Demartini. 2020. Crowd worker strategies in relevance judgment tasks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 241–249.

[32] Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1120–1130.

[33] Jeff Howe. 2006. The rise of crowdsourcing. *Wired magazine* 14, 6 (2006), 1–4.

[34] Kosetsu Ikeda, Atsuyuki Morishima, Habibur Rahman, Senjuti Basu Roy, Saravanan Thirumuruganathan, Sihem Amer-Yahia, and Gautam Das. 2016. Collaborative crowdsourcing with crowd4u. *Proceedings of the VLDB Endowment* 9, 13 (2016), 1497–1500.

[35] Panagiotis G Ipeirotis. 2010. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17, 2 (2010), 16–21.

[36] Ayush Jain, Akash Das Sarma, Aditya Parameswaran, and Jennifer Widom. 2017. Understanding workers, developing effective tasks, and enhancing marketplace dynamics: A study of a large crowdsourcing marketplace. *arXiv preprint arXiv:1701.06207* (2017).

[37] Collins-Thompson Kevyn. 2014. Computational assessment of text readability. *ITL-International Journal of Applied Linguistics* 165, 2 (2014), 97–135.

[38] Shashank Khanna, Aishwarya Ratan, James Davis, and William Thies. 2010. Evaluating and improving the usability of Mechanical Turk for low-income workers in India. In *Proceedings of the first ACM symposium on computing for development*. 1–10.

[39] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Technical Report. Naval Technical Training Command Millington TN Research Branch.

[40] Aniket Kittur, Ed H Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with Mechanical Turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 453–456.

[41] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. 2013. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*. 1301–1318.

[42] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E Kraut. 2011. Crowdforge: Crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 43–52.

[43] Anand Kulkarni, Philipp Gutheim, Prayag Narula, David Rolnitzky, Tapan Parikh, and Björn Hartmann. 2012. Mobileworks: Designing for quality in a managed crowdsourcing architecture. *IEEE Internet Computing* 16, 5 (2012), 28–35.

[44] Anand P Kulkarni, Matthew Can, and Bjoern Hartmann. 2011. Turkomatic: automatic recursive task and workflow design for mechanical turk. In *CHI'11 extended abstracts on human factors in computing systems*. 2053–2058.

[45] Nedim Lipka and Benno Stein. 2010. Identifying Featured Articles in Wikipedia: Writing Style Matters. In *19th International Conference on World Wide Web (WWW 2010)*, Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti (Eds.). ACM, 1147–1148. https://doi.org/10.1145/1772690.1772847

[46] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. 2010. Turkit: human computation algorithms on mechanical turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 57–66.

[47] VK Chaithanya Manam, Dwarakanath Jampani, Mariam Zaim, Meng-Han Wu, and Alexander J. Quinn. 2019. TaskMate: A Mechanism to Improve the Quality of Instructions in Crowdsourcing. In *Companion Proceedings of The 2019 World Wide Web Conference*. 1121–1130.

[48] VK Chaithanya Manam and Alexander J Quinn. 2018. Wingit: Efficient refinement of unclear task instructions. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*.

[49] Brian McInnis, Dan Cosley, Chaebong Nam, and Gilly Leshed. 2016. Taking a HIT: Designing around rejection, mistrust, risk, and workers' experiences in Amazon Mechanical Turk. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. 2271–2282.

[50] Zahra Nouri, Henning Wachsmuth, and Gregor Engels. 2020. Mining Crowdsourcing Problems from Discussion Forums of Workers. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, Barcelona, Spain (Online), 6264–6276. https://doi.org/10.18653/v1/2020.coling-main.551

[51] Alexandra Papoutsaki, Hua Guo, Danae Metaxa-Kakavouli, Connor Gramazio, Jeff Rasley, Wenting Xie, Guan Wang, and Jeff Huang. 2015. Crowdsourcing from scratch: A pragmatic experiment in data collection by novice requesters. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 3.

[52] Alexander J Quinn and Benjamin B Bederson. 2011. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1403–1412.

[53] Niloufar Salehi, Jaime Teevan, Shamsi Iqbal, and Ece Kamar. 2017. Communicating context to the crowd for complex writing tasks. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. 1890–1901.

[54] Thimo Schulze, Stefan Seedorf, David Geiger, Nicolas Kaufmann, and Martin Schader. 2011. Exploring task properties in crowdsourcing–An empirical study on Mechanical Turk. (2011).

[55] M Six Silberman, Joel Ross, Lilly Irani, and Bill Tomlinson. 2010. Sellers' problems in human computation markets. In *Proceedings of the acm sigkdd workshop on human computation*. 18–21.

[56] Henning Wachsmuth and Till Werner. 2020. Intrinsic Quality Assessment of Arguments. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, Barcelona, Spain (Online), 6739–6745. https://doi.org/10.18653/v1/2020.coling-main.592

[57] Daniel S Weld, Christopher H Lin, and Jonathan Bragg. 2015. Artificial intelligence and collective intelligence. *Handbook of Collective Intelligence* (2015), 89–114.

[58] Meng-Han Wu and Alexander James Quinn. 2017. Confusing the crowd: Task instruction quality on amazon mechanical turk. In *Fifth AAAI Conference on Human Computation and Crowdsourcing*.