Quality Control in Crowdsourcing: A Survey of Quality Attributes, Assessment Techniques, and Assurance Actions

FLORIAN DANIEL, Politecnico di Milano
PAVEL KUCHERBAEV, Delft University of Technology
CINZIA CAPPIELLO, Politecnico di Milano
BOUALEM BENATALLAH, University of New South Wales
MOHAMMAD ALLAHBAKHSH, University of Zabol

Crowdsourcing enables one to leverage on the intelligence and wisdom of potentially large groups of individuals toward solving problems. Common problems approached with crowdsourcing are labeling images, translating or transcribing text, providing opinions or ideas, and similar—all tasks that computers are not good at or where they may even fail altogether. The introduction of humans into computations and/or everyday work, however, also poses critical, novel challenges in terms of quality control, as the crowd is typically composed of people with unknown and very diverse abilities, skills, interests, personal objectives, and technological resources. This survey studies quality in the context of crowdsourcing along several dimensions, so as to define and characterize it and to understand the current state of the art. Specifically, this survey derives a quality model for crowdsourcing tasks, identifies the methods and techniques that can be used to assess the attributes of the model, and the actions and strategies that help prevent and mitigate quality problems. An analysis of how these features are supported by the state of the art further identifies open issues and informs an outlook on hot future research directions.

Categories and Subject Descriptors: A.1 [**Introductory and Survey**]; H.3.5 [**Online Information Services**]: Web-based Services; H.1.2 [**User/Machine Systems**]: Human Information Processing

General Terms: Human Factors, Measurement

Additional Key Words and Phrases: Crowdsourcing, quality model, attributes, assessment, assurance

ACM Reference format:
Florian Daniel, Pavel Kucherbaev, Cinzia Cappiello, Boualem Benatallah, and Mohammad Allahbakhsh. 2018. Quality Control in Crowdsourcing: A Survey of Quality Attributes, Assessment Techniques, and Assurance Actions. *ACM Comput. Surv.* 51, 1, Article 7 (January 2018), 40 pages.
https://doi.org/10.1145/3148148

The work of B. Benatallah is supported by the ARC (Australian Research Council) Discovery Project Grant No. DP1601104515: Integrating Quality Control into Crowd-Sourcing Services.

Authors' addresses: F. Daniel and C. Cappiello are with the Politecnico di Milano, DEIB, Via Ponzio 34/5, 20133 Milano, Italy; emails: {florian.daniel, cinzia.cappiello}@polimi.it; P. Kucherbaev is with EEMCS, Web Information System, P.O. Box 5031, 2600 GA Delft, The Netherlands; email: p.kucherbaev@tudelft.nl; B. Benatallah is with the University of New South Wales, CSE, Sydney, NSW 2052, Australia; email: b.benatallah@unsw.edu.au; M. Allahbakhsh is with the University of Zabol, Computer Department, Faculty of Engineering, Zabol, Iran; email: allahbakhsh@uoz.ac.ir.

https://doi.org/10.1145/3148148

1 INTRODUCTION

Crowdsourcing is the outsourcing of a piece of work to a crowd of people via an open call for contributions (Howe 2006). In crowdsourcing, one group of people (so-called *requesters*) submit *tasks* to a crowdsourcing *platform* (or service); another group of people (the *workers* that form the *crowd*) contribute to solving the task. The result of solving the task is called an *output*. Requesters may evaluate outputs and *reward* workers depending on the respective quality; in situations where requesters delegate responsibility for quality control to the crowdsourcing platform, outputs may be checked and rewarded directly and automatically by the crowdsourcing service itself. *Rewards* can be money, gifts, reputation badges, or similar (Minder and Bernstein 2012).

Depending on the task to crowdsource and the acceptance criteria by both the requester and the workers to enter a mutual business relationship, different negotiation models have emerged so far: The *marketplace model* (Ipeirotis 2010) targets so-called micro-tasks of limited complexity, such as tagging a picture or translating a piece of text, for which the requester typically requires a large number of answers. Prominent examples of crowdsourcing platforms that implement the marketplace model are Amazon Mechanical Turk, Microworkers, and CrowdFlower. The *contest model* (Cavallo and Jain 2012) is particularly suitable to creative works where the requester fixes the budget he is willing to spend and workers compete with their solutions for the reward. Examples are 99designs, InnoCentive, and IdeaScale. The *auction model* (Satzger et al. 2013) targets works where the requester fixes the acceptance criteria and workers bid for the task. An example of an auction platform is Freelancer. But also *volunteering*, e.g., like in Wikipedia or Crowdcrafting, has proven its viability, and the spectrum of variations of these models is growing.

The critical aspect is that outputs produced by the crowd must be checked for *quality*, since they are produced by workers with unknown or varied skills and motivations (Minder and Bernstein 2012; Malone et al. 2010). The quality of a crowdsourced task is multifaceted and depends on the quality of the workers involved in its execution, the quality of the processes that govern the creation of tasks, the selection of workers, the coordination of sub-tasks like reviewing intermediary outputs, aggregating individual contributions, and so on. A large body of empirical studies confirms that existing crowdsourcing platforms are not robust to effectively check and control the quality of crowdsourced tasks or to defend against attacks such as cheating, manipulating task outputs, or extracting sensitive information from crowdsourcing systems (Kritikos et al. 2013). Concerns about the unintended consequences of poor quality control methods, including financial, intellectual property and privacy risks, malicious attacks, and project failure are growing (Minder and Bernstein 2012; Malone et al. 2010; Kritikos et al. 2013).

To be fair, it is important to note that platform providers may not have the necessary information about tasks to approach these concerns appropriately. For instance, in marketplace platforms task design is typically fully under the control of the requester, and the platform is not aware if the task raises intellectual property issues or not. Some quality control and submission filtering activities may therefore also be carried out by the requester outside the platform to meet expected levels of quality. Task-specific platforms, such as 99 designs for graphical design tasks, instead, are well aware of the problem (e.g., intellectual property rights) and help requesters and workers to manage them. They can do so, as they focus on few specific task types only, which they know and can support well.

The increasing importance of crowdsourcing services and the intensification of global competition indicate, however, that building proper solutions to quality problems should now be a top priority. Ideally, developers of crowdsourcing applications should be offered effective quality control techniques, methods, and tools that let them systematically build, verify, and change quality control mechanisms to suit their requirements. Yet, no framework exists that endows crowdsourcing services with robust and flexible quality control mechanisms, and most research

on quality control in crowdsourcing has focused on single, specific aspects of quality only, such as worker reputation or redundancy. In addition, conceived quality control techniques are typically embedded inside proprietary platforms and not generalizable. Consequently, designing, building, and maintaining robust and flexible crowdsourcing quality controls remains a deeply challenging and partly unsolved problem, and requesters and platform operators may not have the knowledge, skills, or understanding to craft their own quality-aware strategy to safely take advantage of the opportunities offered by crowdsourcing. For instance, Stol and Fitzgerald (2014) describe a case study on crowdsourced software development that shows how unpreparedness to handle quality issues quickly increased project costs. Lasecki et al. (2014) show how state-of-the-art crowd-powered systems are still not ready to deal with "active attacks," while Gadiraju et al. (2015) show that malicious workers can easily cause harm if suitable quality controls are missing.

This survey aims to shed light on the problem of quality control in crowdsourcing and to help users of crowdsourcing services and developers of crowdsourcing applications to understand the various moments where quality comes into play in the crowdsourcing process, how it is manifest (or not), and how it can be assessed and assured via suitable methods and actions. Concretely, it provides the following contributions:

- An *introduction* to quality control in crowdsourcing (Section 2) and a *taxonomy* to understand and classify the state of the art in quality control techniques for crowdsourcing. The taxonomy focuses on the three core aspects of quality:
  - The *quality model* that emerges from the state of the art (Section 3), that is, the dimensions and attributes to describe quality in crowdsourcing services.
  - The *assessment* (the measuring and its methods) of the values of the attributes identified by the quality model (Section 4). In order to instantiate the quality model, it is necessary to know and master the respective assessment methods.
  - The *assurance* of quality (Section 5), that is, the set of actions that aim to achieve expected levels of quality. To prevent low quality, it is paramount to understand how to design for quality and how to intervene if quality drops below expectations.
- A comprehensive *analysis* of how state-of-the-art crowdsourcing platforms and services support quality control in practice (Section 6).
- A discussion of shortcomings and limitations along with the respective *challenges and opportunities* for future research and development (Section 7).

2 QUALITY IN CROWDSOURCING

Quality (and its control) is an active research area in several computing disciplines, including data quality (Batini et al. 2009; Hunter et al. 2013), quality of software products (Herbsleb et al. 1997; Jung et al. 2004), software services (Kritikos et al. 2013), and user interfaces (Hartmann et al. 2008). In many cases, research from these fields is complementary to quality control in crowdsourcing, and some elements can be adopted to leverage application-specific quality control mechanisms (e.g., some code quality metrics can be leveraged in programming crowdsourcing tasks). However, the peculiarities of crowdsourcing require taking into account novel quality issues that are rising, especially to consider the user involvement in the execution of different tasks (e.g., production of contents, annotations, and evaluations).

Therefore, it is necessary to consider that in crowdsourcing systems the quality of the output is influenced by multiple factors (see Figure 1): The requester defines and publishes tasks to be executed by the crowd via a crowdsourcing platform or a dedicated application. The crowd produces outputs that are checked for quality and delivered to the requester. The output quality

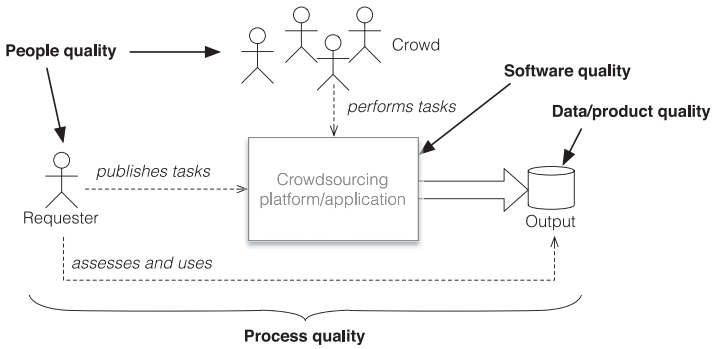

Fig. 1. The basic crowdsourcing scenario with its quality aspects.

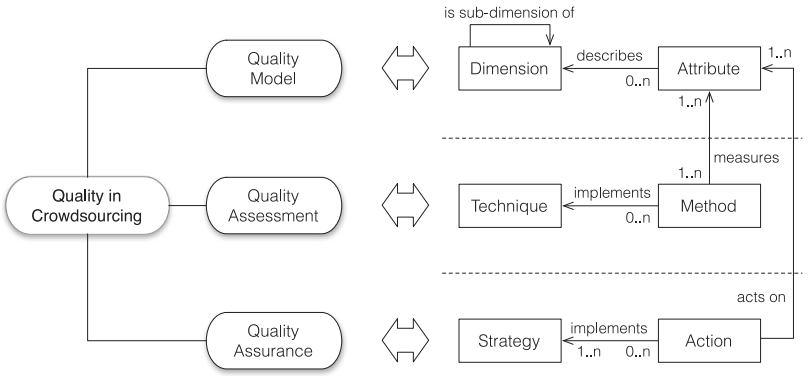

Fig. 2. The components of the quality control taxonomy and their internal structures.

depends on, among other things, the profiles and abilities of workers, the description of the tasks, the incentives provided, the processes implemented to detect malicious behaviors or low-quality data, as well as on the reputation of and collaboration by the requester. The quality of the output can therefore be expressed by means of quality dimensions that model objective and subjective aspects of *data and product quality*, while the actual quality of the outputs is influenced by aspects related to *people* (the workers, requesters, and other possible actors), *software* (the crowdsourcing platform or application and the design of tasks), and *process quality* (the organization of work and the implemented quality measures).

Adapting the expectation-centric definition of quality (e.g., Lewis and Booms (1983)) to crowdsourcing, we define the quality of a crowdsourced task as *the extent to which the output meets and/or exceeds the requester's expectations.*

2.1 Taxonomy

Considering the current fragmented literature and lack of an all-encompassing view of quality control in crowdsourcing, we developed the taxonomy depicted in Figure 2 to understand and analyze how quality is dealt with in crowdsourcing. The proposed taxonomy features a holistic view on the problem of quality control in crowdsourcing and aims to highlight the key aspects and options one has to face when developing quality control mechanisms. The taxonomy is based on our own experience (Allahbakhsh et al. 2013; Cappiello et al. 2011; Kritikos et al. 2013; Batini et al.

2009) as well as on an extensive literature review of related areas, discussions with colleagues, experimentation with systems and prototypes, which allowed us to identify common building blocks for the different variations in quality control mechanisms. Accordingly, we propose a taxonomy with three categories that may, in turn, be split into sub-categories:

- A *quality model* for crowdsourcing that captures which quality dimensions and attributes have been identified so far in the literature.
  - *Dimensions* represent the components that constitute a crowdsourcing task, such as the input and output data of the task or the people involved in the task. Dimensions are described by attributes and are not directly measurable.
  - *Attributes* characterize properties (qualities) of the task, such as the accuracy of data or the expertise of workers. Attributes are *concrete* if they are measurable; they are *abstract* if they are not directly measurable and values are derived from concrete attributes (e.g., aggregations).
- An analysis of the *quality assessment* methods that have been used so far to assess quality attributes in the context of crowdsourcing.
  - *Techniques* distinguish who performs the assessment from a high level of abstraction. For instance, techniques that involve single individuals (e.g., rating) differ from techniques that involve groups of people (e.g., voting).
  - *Assessment methods* allow one to measure quality attributes. For example, accuracy may be measured comparing outputs with a ground truth, while worker expertise may be measured through questionnaires. Methods are like functions that are executed automatically, manually, or both and produce a value as output.
- A study of the *quality assurance* actions that allow one to improve quality by acting on the quality attributes identified in the quality model.
  - *Strategies* represent the top-level decisions to be taken when aiming at improving quality, that is, what to act upon and in which direction. For instance, selecting good workers and training workers are two different strategies that aim to improve the quality of the people involved in a task.
  - *Actions* are the basic operations one can perform to prevent or fix quality problems, such as checking credentials or showing a training video. Each action implements a specific strategy.

In the following sections, we detail each of these components and explain them with help from the respective literature.

2.2 Literature Selection

In order to identify the references to consider in this survey, we selected a set of conferences and journals that, to the best of our knowledge, publish research on crowdsourcing and related topics. The conferences considered were AAAI, BPM, CAiSE, CHI, CI, CIKM, CSCW, ECML, ECSCW, HCOMP, ICML, ICSE, ICWE, iUI, KDD, NIPS, SIGIR, UBICOMP, UIST, VLDB, WSDM, and WWW. The journals considered were ACM CSUR, ACM TIIS, ACM TOCHI, ACM TOIS, ACM TOIT, ACM TOSEM, ACM TWEB, Communications of the ACM, CSCW, Information Systems, IEEE Computer, IEEE Internet Computing, IEEE TKDE, IEEE TSC, IEEE TSE, VLDB, and WWW. In order to keep the selection of references manageable and up to date, we queried for contributions from 2009 onward using the following keywords in either the title, abstract, or keywords: Crowd, Crowdsourcing, Human Computation, Collective Intelligence, Social Computing, Collaborative Computing, Collaborative Systems, Wikinomics, Mass Collaboration, Micro-tasking, Crowd Labour. Articles were retrieved through the advanced search feature of the ACM Digital Library and SCOPUS. Papers

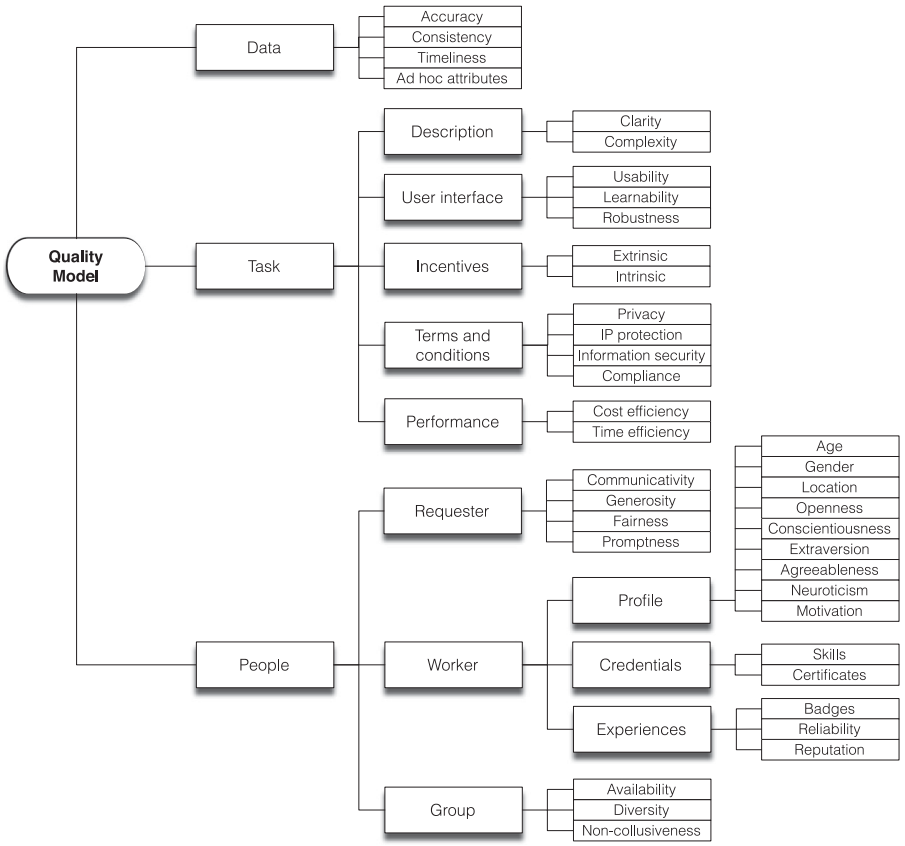

Fig. 3. The quality model for crowdsourcing tasks emerged from literature with *dimensions* (boxes with shadow) and *attributes* (boxes without shadow).

published in HCOMP and Collective Intelligence were retrieved manually, as at the time of querying they were not properly indexed by any digital library. The selection specifically looked for conference and journal papers, and neglected demo papers, posters, and workshop papers. The search identified 1,013 papers. A further manual check filtered out 257 papers that we finally considered relevant for this survey. Additional papers considered stem from prior knowledge by the authors.

3 CROWDSOURCING QUALITY MODEL

Figure 3 illustrates the quality model identified as a result of this survey. We identify the following dimensions to group the quality attributes in crowdsourcing systems:

— *Data*: This refers to the data required to perform a task or produced as a result of performing a task by a worker (i.e., task input and output data). For instance, input data can be images to label or a text to translate, and the corresponding output data can be the labels of the images and the translated text. Quality control in crowdsourcing all revolves around achieving high-quality output data, which is the core challenge for mass adoption of crowd work (Kittur et al. 2013).

— *Task*: The type of work proposed, the way it is described, and how it is implemented strongly affects the attractiveness of a task and the quality of the outputs that can be achieved. We distinguish the following sub-dimensions:
  — *Description*: This is the description of the work the requester asks the crowd to perform; it includes the instructions of how to perform the task and possible context information. The clarity and details of the description influence the way workers perform the task and hence the quality of its output (Chandler et al. 2013).
  — *User interface*: This is the software user interface workers use to perform the task. In most tasks, this interface is a simple HTML form through which workers submit their contributions. But it can also come in the form of a stand-alone application with input techniques not supported by standard HTML forms (e.g., selecting patterns in a model). The quality of the user interface determines how easily workers can perform tasks (Marcus et al. 2012).
  — *Incentives*: Crowdsourcing generally implies paid, online work. The incentives, that is, the stimuli the requester offers to attract workers to tasks, plays therefore a crucial role in crowdsourcing. Incentives may come in two different forms: extrinsic incentives (e.g., rewards) and intrinsic incentives (e.g., worker status). Several studies have identified direct relations between incentives and output quality and/or task execution speed (Singer and Mittal 2013).
  — *Terms and conditions*: These are the general rules and arrangements that govern the work relationship between the requester and the workers. Aspects like the privacy of workers, the protection of intellectual property (IP), compliance with laws, ethical standards, and similar are typically specified here. Terms and conditions affect worker interest and legal aspects (Wolfson and Lease 2011).
  — *Performance*: Performance expresses the amount of useful work completed in the context of a crowdsourced task compared to the time and resources needed for its processing. Crowdsourcing can be an intricate endeavor, with large volumes of input data to be processed and hundreds or thousands of workers involved, that can easily grow complex. Controlling the consumption of resources is of utmost importance to guarantee the sustainability of crowdsourcing initiatives.
— *People*: These are the actors involved in the task, including requesters, workers, and the crowd. Good work requires qualified, prepared and trusted people. In some tasks, workers may also be required to collaborate with other workers or requesters. We distinguish three sub-dimensions regarding the people involved in crowdsourcing:
  — *Requester*: The first sub-dimension acknowledges the role of the requester, who is not only the one that crowdsources work, but also the one that may evaluate outputs and interact with workers. Fairness and communication are examples of attributes of a good requester (Irani and Silberman 2013; Allahbakhsh et al. 2012).
  — *Worker*: The other natural sub-dimension focuses on the workers. Substantial work has been done on discriminating good from bad workers. We group the identified attributes into profile, credentials, and experiences, in function of what the attributes describe. *Profile* characterizes an individual's identity (age, gender, location) and describes the individual's distinctive character, typically expressed through motivation and so-called personality traits, that is, relatively stable, enduring properties of people that can influence behavior (John et al. 2008). *Credentials* are documented qualifications that people bring into a system from the outside. *Experiences* express knowledge acquired or social relations established by using a system; they are usually based on system-specific metrics and rules.
  — *Group*: Finally, people can be studied also in groups, for example, the crowd as a whole or smaller teams of collaborating workers. Good groups, for instance, are formed by

non-colluding workers (KhudaBukhsh et al. 2014) or by workers that are able to perform also very diverse tasks with good quality.

We describe the attributes that characterize each dimension next. Appendix A summarizes the referenced literature.

3.1 Data Quality

A large body of work focused on characterizing and handling data quality in the context of crowdsourcing. The most important attribute studied is the *accuracy* of output data (Hansen et al. 2013; Kazai et al. 2011a; Yin et al. 2014). Several synonyms are used to refer to accuracy, such as "goodness" (Cao et al. 2014), "correctness" (Yin et al. 2014), or just generically "quality" (Eickhoff et al. 2012; Kulkarni et al. 2012a; Kern et al. 2010). Data *consistency* is commonly interpreted as the similarity between outputs produced by different workers in response to a same input. It has been studied, for instance, in the context of peer consistency evaluation (Huang and Fu 2013b; Eickhoff et al. 2012). The *timeliness* of data is the property of outputs to be available in useful time for further processing (Kittur et al. 2013). Timeliness is especially important in near-real-time crowdsourcing scenarios (Lasecki et al. 2013a and 2013b) and has also been studied under the name of "reaction time" (Yin et al. 2014).

It is important to note that next to these quality attributes that characterize the quality of a dataset in general terms, many crowdsourcing scenarios ask for the use of *ad-hoc attributes* that are able to capture task-specific properties. For instance, Yu et al. (2014) study the accuracy, coverage, and conciseness of textual summaries; the latter two attributes are specific to the problem of summarizing texts. Nguyen et al. (2014) distinguish between 13 different features to assess narrative similarity.

3.2 Task Description Quality

Regarding the quality of task descriptions, its *clarity* is of utmost importance (Hossfeld et al. 2014). Tokarchuk et al. (2012) state that clarity positively correlates with performance, a result that is empirically confirmed by Georgescu et al. (2012), while Kulkarni et al. (2012b) specifically study the problem of incomplete descriptions. Needless to say, if a worker does not immediately understand a task, he will not be willing to work on it. Several authors have studied the *complexity* of tasks (or "granularity" (Hu et al. 2012)), identified correlations with worker motivation (Rogstadius et al. 2011), and matched workers with tasks based on complexity scores (Difallah et al. 2013).

3.3 User Interface Quality

A user-friendly, understandable interface can attract more workers and increase the quality of outputs (Allahbakhsh et al. 2013). In this respect, especially *usability* has been studied, for example, for workers from low-income countries (Khanna et al. 2010), in photographing tasks (Noronha et al. 2011), or in task design (Retelny et al. 2014). Alagarai Sampath et al. (2014) specifically focus on visual saliency and working memory requirements as sub-properties of usability. Willett et al. (2012) show that the design of the task interface has an impact on the *learnability* of a task. A good task interface is further characterized by a high *robustness* against cheaters, that is, it is able to produce high-quality outputs even in the presence of cheaters (Eickhoff et al. 2012). Hung et al. (2013a) talk about "sensitivity to spammers."

3.4 Incentives

Incentives affect the attractiveness of a task. As Hossfeld et al. (2014) point out, there are several possible incentives targeting either the extrinsic (reward-driven) or intrinsic (interest-driven)

motivation of workers. According to the authors, increasing extrinsic motivation leads to faster task completion, while increasing intrinsic motivation leads to higher quality. Many researchers have specifically studied the role of monetary rewarding in crowdsourcing, for example, on speed (Heer and Bostock 2010) or execution efficiency (Singer and Mittal 2013), while Eickhoff et al. (2012) compared fun and social prestige with monetary rewards as incentives.

3.5 Terms and Conditions

At a more abstract level, *privacy* has been identified as the key property of tasks that deal with personal data, for example, using images that show people (Lasecki et al. 2013) or ask workers to share their position (Boutsis and Kalogeraki 2016). But also *information security* and *IP protection*, that is, the protection of data and IP, are emerging quality attributes that affect a requester's willingness to crowdsource (Vukovic and Bartolini 2010). A requester may, for example, share source code or design documents, which are assets that contain IP; quality control mechanisms are needed, for example, to limit the access of workers to information, invite vetted workers only, or sign nondisclosure agreements. As Amor et al. (2016) point out in their approach to crowdsourced team competitions, the problem is not limited to requesters only and may also affect workers. More generically, *compliance* means conformance with laws and regulations (Wolfson and Lease 2011), but also with commonly accepted user policies (Wang et al. 2012) (e.g., no malicious crowdsourcing campaigns) or expected ethical behaviors by the requester (Irani and Silberman 2013). If a task is perceived as non-compliant by workers, it is unlikely they will perform it.

3.6 Task Performance

The two most important attributes that have been studied to capture the execution performance of a task are cost efficiency and time efficiency. The expected cost of a task is easily determined by multiplying the reward by the number of task instances worked on Ambati et al. (2012) and Livshits and Mytkowicz (2014). However, since crowdsourcing a task is a generally non-deterministic process, *cost efficiency*, that is, the cost per completed task instance or the cost per correct output, has been studied more intensively (Ipeirotis and Gabrilovich 2014; Rokicki et al. 2014). The *time efficiency* can be defined as the number of tasks completed in a given temporal interval (Eickhoff et al. 2012); Lin et al. (2014) use the synonym "throughput" and Hung et al. (2013b) talk about "computation time" for a given set of tasks. Kucherbaev et al. (2016a), for instance, aim to improve the time efficiency by re-launching tasks at runtime. Cheng et al. (2015a) compute task effort based on error rates and task completion times.

3.7 Requester Reputation

Irani and Silberman (2013) propose Turkopticon, a browser extension for Firefox and Chrome that augments workers' view of their task list in Amazon Mechanical Turk with information other workers have provided about requesters. Turkopticon supports assessing requesters by means of four attributes: *Communicativity* captures how responsive a requester is to communications or concerns raised by a worker. *Generosity* tells how well a requester pays for the amount of time necessary to complete a task. *Fairness* (also studied by Allahbakhsh et al. (2012)) tells how fair a requester is in approving or rejecting work submitted by workers. *Promptness* captures how promptly a requester approves and pays work that has been successfully submitted.

3.8 Worker Profile

Kazai et al. (2011b) study different worker profile attributes and personality traits in labeling tasks. *Age*, for instance, correlates with output accuracy, while *gender* does not seem to have any influence on quality. *Location* does impact quality (Kazai et al. 2011b; Eickhoff et al. 2012; Kazai

et al. 2012). To assess the personality of workers, Kazai et al. (2011b) use the so-called Big Five inventory and study five personality traits (definitions by John et al. (2008) and correlations from Kazai et al. (2011b)): *Openness* "describes the breadth, depth, originality, and complexity of an individual's mental and experiential life"; it correlates with output accuracy. *Conscientiousness* "describes socially prescribed impulse control that facilitates task- and goal-directed behavior, such as thinking before acting"; it positively correlates with output accuracy. *Extraversion* "implies an energetic approach toward the social and material world and includes personality traits such as sociability, activity, assertiveness, and positive emotionality"; it negatively correlates with output accuracy. *Agreeableness* "contrasts a prosocial and communal orientation toward others with antagonism and includes traits such as altruism, tender-mindedness, trust, and modesty"; it correlates with output accuracy. *Neuroticism* "contrasts emotional stability and even-temperedness with negative emotionality, such as feeling anxious, nervous, sad, and tense"; it negatively correlates with output accuracy. According to Kazai et al. (2012), personality characteristics are useful to distinguish between good and better workers (less between good and bad). Kobayashi et al. (2015) introduce a taxonomy of worker *motivations*, so that appropriate incentives could be applied to persuade workers to contribute.

3.9 Worker Credentials

Credentials are all those qualifications or personal qualities that describe a worker's background; they can be self-declared (e.g., programming language skills) or issued by official bodies (e.g., a MSc degree is issued by a university). *Skills* are abilities that tell if a worker is able to perform a given task. Mavridis et al. (2016) propose a taxonomy of skills; Difallah et al. (2013) use skills to match workers and tasks; Schall et al. (2014) identify skills automatically. *Certificates* are documents that attest skills, such as academic certificates or language certificates (Allahbakhsh et al. 2013).

3.10 Worker Experience

Badges are platform-provided certificates of performance, for example, performing a certain number of actions or tasks of a given type (Anderson et al. 2013). Badges can be used to select workers, but also to motivate them: badges are seen as "virtual trophies" (Scekic et al. 2013a). The *reliability* of a worker (often also called "accuracy" of the worker) is commonly interpreted as the aggregated accuracy of the outputs produced by the worker (Kazai et al. 2011a), or the worker's error rate in answering questions (Dalvi et al. 2013; Demartini et al. 2013), or the acceptance rate of outputs delivered by workers (e.g., Mechanical Turk). Sakurai et al. (2013) use the synonym "correctness" of a worker as the probability of the worker being correct in labeling tasks. Raykar and Yu (2011) assign scores to workers based on how reliable (not random) their answers are. The *reputation* of a worker may take into account additional parameters, such as the worker's timeliness, the quality of the evaluators that assessed the worker, relations with other workers or requesters, the trust they express toward the worker, and similar (Allahbakhsh et al. 2013). That is, reputation also captures other community members' feedback about a worker's activity in the system (De Alfaro et al. 2011).

3.11 Group Quality

The quality of groups of people, that is, teams or the crowd as a whole, has been studied mostly in terms of three different aspects. *Availability*, that is, the presence in a platform of enough workers or experts with the necessary skills for a given task, has been identified as an issue (Li et al. 2014). Low availability usually leads to low quality of outputs (Ambati et al. 2012) or slow task execution (Li et al. 2014). Next, *diversity* is the property of a group to represent different types of people, skills, opinions, and similar. Diversity is particularly important if representative samples of people are searched for, for example, in tasks that ask for opinions like the ox weight estimation experiment

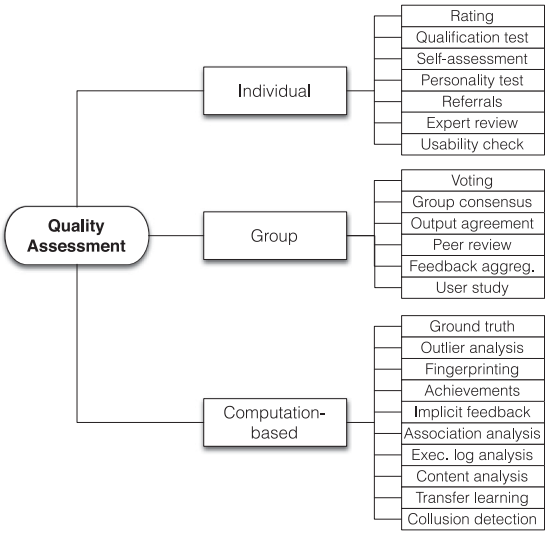

Fig. 4. The quality assessment *methods* (on the right) emerged from literature; the boxes with shadow represent generic *techniques*.

by Surowiecki (2005) or polls (Livshits and Mytkowicz 2014). Willett et al. (2012) specifically study how to increase the diversity of worker outputs. Finally, *non-collusiveness* means that a group of agents does not share information or make joint decisions contrary to explicit or implicit social rules, which would result in an unfair advantage over non-colluding agents or other interested parties (KhudaBukhsh et al. 2014).

4 QUALITY ASSESSMENT

Assigning concrete values to some of the attributes identified above can be a straightforward exercise, for example, verifying if a worker has the necessary skills, certificates or badges for a task can easily be done manually or automatically (Khazankin et al. 2012), or a task duration can easily be read from the log of the crowdsourcing system. Assessing some other attributes may instead require the use of methods based on both automated metrics as well as manual interventions by workers, the requester, and/or external experts. In the following, we focus on these more complex methods, and organize the discussion of the identified assessment methods into three major groups (the techniques) in function of the actor in the assessment task, as illustrated in Figure 4:

— *Individual*: Given the human-oriented nature of crowdsourcing, it is natural to think about involving humans also into assessment tasks and not only into work tasks. Some assessment methods require the involvement of individuals (workers, experts, or the requester), such as rating the accuracy of a given output or writing a review.
— *Group*: Some other assessment methods require the joint action of a group of people (typically workers) for the formation of an assessment. For instance, voting requires multiple participants to derive a ranking from which to select the best, or peer review requires multiple peers to judge the work of a colleague.
— *Computation-based*: Some other assessment methods, instead, can be performed without the involvement of humans, that is, automatically by a machine. Comparing a set of outputs

with a given ground truth can, for example, be carried out automatically if suitable comparison operators are available.

We describe the respective methods next, following the order proposed in Figure 4. We summarize the discussion tabularly in Appendix A.

4.1 Individual Assessments

4.1.1 Rating. Rating means assigning a value to an item chosen from a scale to express the perceived quality of the item. The scale defines the possible values to choose from, for example, unary scales allow one to express if an item is liked or not (used, for instance, in Instagram), binary scales distinguish between two values (good/bad, true/false, thumbs-up/thumbs-down, or similar), ordinal scales distinguish between discrete sets of positive and negative values (typically 5, 7, or more), and continuous scales may allow an arbitrary number of values inside an interval ([0..1], [1..100], or similar). Rating is extensively used in crowdsourcing for quality assessment (Dalvi et al. 2013; Yu et al. 2014), text similarity and understandability (Nguyen et al. 2014), worker confidence (Sakurai et al. 2013), task quality (in CrowdFlower workers rate tasks after completion), and requester reputation (Irani and Silberman 2013). Hata et al. (2017) have demonstrated that it is possible to predict workers' long-term quality using just a glimpse of their quality on the first five tasks as rated by the requester. Instead, to counteract reputation inflation (too generous ratings, even if the rater would not want to work with the rated worker/requester again), Gaikwad et al. (2016) conceived a rating system that rebounds the consequences of feedback back onto the raters, for example, by giving positively rated workers precedence in future tasks by the same requester.

4.1.2 Qualification Test. A qualification test is a questionnaire that a worker may be required to fill out to obtain access to tasks that require prior knowledge or skills, for example, language skills or acquaintance with a programming or modeling tool. Thanks to the *a priori* knowledge of the correct answers of the test, qualification tests can be evaluated automatically, granting the worker access to the task if a minimum threshold of correct answers is reached. Qualification tests are widespread in crowdsourcing and many crowdsourcing platforms provide support for qualification tests. Heer and Bostock (2010) studied the effectiveness of qualification tests to assess workers in graphical perception experiments; their findings show that qualification tests are indeed able to discriminate unprepared workers, thereby increasing output quality.

4.1.3 Self-Assessment. Self-assessment asks workers to assess the quality of their own work after producing an output for a task. The practice stems from self-assessment in learning (Boud 2013), where the aim is to help students reflect, learn, and connect their work better with learning goals. Dow et al. (2012) studied the effectiveness of self-assessment in the context of crowdsourced product reviews and found that tasks with self-assessment produced better overall quality than tasks without. In addition, they found that self-assessment helped workers improve the quality of their work over time, in line with the expected learning effect. Self-assessment has also been used to allow workers to state the confidence they have in their own work (Sakurai et al. 2013), even suggesting workers skip the task if they are not confident (Shah and Zhou 2015). Shah and Zhou (2016) show noisy output examples to workers right before task submission so they can resolve possible "silly" mistakes.

4.1.4 Personality Tests. These are tests, typically questionnaires shown to workers, that allow one to assess personality traits, that is, the actions, attitudes, and behaviors individuals possess. For instance, Kazai et al. (2011b, 2012) used the so-called Big Five inventory (John and Srivastava 1999) to assess the personality traits openness, conscientiousness, agreeableness, extraversion, and neuroticism; the authors further identified a positive relation between the former three

and output accuracy and a negative relation between the latter two and output accuracy. Turkopticon proposed by Irani and Silberman (2013), instead, allows workers to explicitly rate the communicativity, generosity, fairness, and promptness of requesters.

4.1.5 Referrals. Referrals express that someone has referred someone else for consideration. They are well known in the domain of recruitment as a way to gather and confirm expertise. SalesForce has indicated in their official blog that their main strategy for recruitment is based on referrals (https://www.salesforce.com/blog/2015/01/behind-scenes-salesforce-our-1-recruiting-secret.html). LinkedIn and ResearchGate allow tagging skills to professionals. While referrals may not be a common instrument to identify workers, they may well be used to find experts. Facebook and Twitter enable recruiting workers by exploiting social connections (Bozzon et al. 2012).

4.1.6 Expert Review. An expert review is an assessment provided by a person that is considered a domain expert by the requester. This expert is commonly not a member of the crowd (whose work the expert assesses) and is typically directly assigned by the requester to assessment tasks. Dow et al. (2012) describe a system that supports expert reviews providing workers with feedback on the quality of their work and show that expert reviews help increase quality. Expert reviews are, however, costly; accordingly, Hung et al. (2015) devised a method that optimizes the effort of experts through reviews of only partial worker output sets while keeping quality high.

4.1.7 Usability Check. Checking the usability of a task design helps identify issues that may prevent workers from producing high-quality outputs. Specifically, usability guidelines (Nielsen et al. 2002) can be used by the requester to check if a task design follows known best practices or not. In the specific context of crowdsourcing, Willett et al. (2012) have studied the usability of task UIs (understandability and learnability) for data analysis and identified seven guidelines for data analysis tasks: use feature-oriented prompts, provide good examples, include reference gathering subtasks, include chart reading subtasks, include annotation subtasks, use pre-annotated charts, and elicit explanations iteratively.

4.2 Group Assessments

4.2.1 Voting. Voting means expressing preference for one or more candidates (e.g., outputs) out of a group of candidates; aggregating the votes by multiple voters enables the derivation of a list of candidates ranked by preference. Voting is very common in crowdsourcing and used to make group decisions, for example, two out of three majority decisions. Kulkarni et al. (2012a), for instance, provide built-in support for voting in their collaborative crowdsourcing platform Turkomatic to validate the quality of task outputs. Little et al. (2010a) instead equip their human programming framework Turkit with a dedicated *vote* programming construct. Caragiannis et al. (2014) study different voting techniques for crowdsourcing, while Sun and Dance (2012) also point out some pitfalls of voting that should be taken into account when using the technique.

4.2.2 Group Consensus. Group consensus is similar to voting, yet, the consensus refers to ratings assigned to an item and less to a mere expression of preference. The purpose is therefore not ranking multiple items (outputs) but identifying the most representative rating for one item. The method is frequently used to produce consensus labels from crowd data. Sheshadri and Lease (2013) compare different techniques to assess offline consensus and study, given multiple noisy labels (or ratings) per item, how to infer the best consensus label. Zhang et al. (2015) specifically address the problem of imbalanced labeling, that is, labeling scenarios where there are many more positive than negative cases (or vice versa). Eickhoff et al. (2012) use disagreement with the majority consensus to identify workers considered cheaters (if they disagree more than 67% of the times with the majority) and to derive a measure of robustness of a task. However, consensus

must not always be good: Kairam and Heer (2016), for instance, show that there is also value in divergent outputs in crowdsourcing.

4.2.3 Output Agreement. An output agreement is reached if two or more workers, given the same input and performing the same task, produce a same or similar result as output. Waggoner and Chen (2014), for instance, study the use of output agreement to assess worker reliability and show that the technique is particularly useful to elicit common knowledge (since the workers know that they are assessed based on the similarity of their outputs with those by other workers). Huang and Fu (2013b) use output agreement (they use the term peer consistency) to assess and motivate workers. Jagabathula et al. (2014) instead assess workers based on output disagreement.

4.2.4 Peer Review. Peer review is similar to expert reviews, with the key difference that it involves multiple peers in the assessment, in order to limit the bias of individual peers and to elicit an as correct as possible assessment. It is typically used in those situations where experts would not be able to assess alone all outputs or items, such as in the paper selection of scientific conferences, and represents a form of community-based self-assessment. Crowdsourcing shares similar characteristics. Hansen et al. (2013), for example, use peer review in the context of the FamilySearch Indexing project and show that peer review is more efficient than using an arbitrator to resolve disagreements among volunteers. Zhu et al. (2014) study different peer reviewing approaches and show that the practice, next to representing an effective assessment instrument, may also lead to performance increases of the reviewers (the workers themselves). Peer review among workers has also been successfully used as a reputation assessment technique, producing better results than conventional techniques (Whiting et al. 2017).

4.2.5 Feedback Aggregation. More sophisticated aggregation algorithms can be used to integrate large amounts of feedbacks provided by either workers or requesters, in order to obtain representative, concise assessment of quality attributes. Dalvi et al. (2013), for instance, propose an eigenvector-based technique to estimate both worker reliabilities and output qualities. Simple analytic models (e.g., sum, average, minimum, or maximum) are employed to calculate the rating scores of products. In weighted averaging techniques, evaluations cast by users are weighted and their impact on final rating scores is adjusted based on their corresponding weights. Davtyan et al. (2015) leverage on content similarity to increase the quality of aggregated votes compared to standard majority voting approaches. Allahbakhsh et al. (2012) use time and credit of crowdsourcing tasks to weight pairwise evaluations between the community members and also propose the concept of fairness of an evaluator while evaluating other members. Iterative approaches calculate weights of evaluations and the result of aggregation simultaneously but in several iterations. Iterative methods for social rating have been pioneered in Laureti et al. (2006) and Yu et al. (2006). Ignjatovic et al. (2008) have proposed a reputation calculation model for online markets. Many more examples of aggregation algorithms exist (Hung et al. 2013b), with Joglekar et al. (2013) also generating confidence intervals for aggregated values.

4.2.6 User Study. Assessing the effectiveness of task UIs for which there do not yet exist reference guidelines may require conducting task-specific usability studies. Willett et al. (2012), for instance, conducted user studies directly on Amazon Mechanical Turk without direct contact between the experimenters and the workers. Khanna et al. (2010), instead, organized controlled between-subjects study sessions with workers, implemented dedicated image annotation tasks, and observed them in action to assess the usability of tasks to low-income workers from countries like India. The key general barriers identified were the complexity of instructions and UIs, difficult navigation, and different cultural backgrounds. Alagarai Sampath et al. (2014) used eye tracking

to measure the working memory required by different task designs and their visual saliency. This kind of user study requires expert skills, for example, for the conduct of interviews, the design of questionnaires, or proper user observations.

4.3 Computation-based Assessments

4.3.1 Ground Truth. The use of ground truth data (gold data, control questions) is a common approach in crowdsourcing: by injecting into tasks questions whose answers are known and formalized *a priori* (so that they can be checked automatically), it is possible to computationally estimate the aggregate accuracy of outputs and the trust in workers. Ground truth evaluation is considered one of the most objective mechanisms that can accurately measure the performance of workers (Huang and Fu 2013b). The method is, for instance, natively supported by CrowdFlower. Eickhoff et al. (2012) provide an example of how to use ground truth data in gamification. Hara et al. (2013) provide good examples of how to collect ground truth answers for image labeling with wheelchair drivers (domain experts), while Oleson et al. (2011) argue that the generation of ground truth data is very difficult and costly and propose the use of "programmatic gold" generated from previously collected correct data. Liu et al. (2013) propose a method predicting an optimal number of ground truth labels to include. Le et al. (2010) study the distribution logics of gold questions and conclude that a uniform distribution produces best results in terms of worker precision, while El Maarry et al. (2015) show that the biggest threat to ground truth evaluations are tasks with highly skewed answer distributions. CAPTCHAs can be used to tell human workers and machines (e.g., robots) apart (Lasecki et al. 2014; Von Ahn et al. 2008).

4.3.2 Outlier Analysis. Outliers are data points (e.g., worker performance or estimations of a property) that significantly differ from the remaining data (Aggarwal 2013), up to the point where they raise suspicion. Outliers may thus identify poorly performing workers, random answers, or similar. In Rzeszotarski and Kittur (2012), the authors show how they use CrowdScape to link behavioral worker information with output properties and identify outliers visually (in charts). Jung and Lease (2011) use outlier analysis to identify "noisy workers" in consensus labeling tasks. Marcus et al. (2012) use outlier analysis to detect "spammers" in estimation tasks.

4.3.3 Fingerprinting. This method captures behavioral traces from workers during task execution and uses them to predict quality, errors, and the likelihood of cheating. Behavioral traces are identified by logging user interactions with the task UI (at the client side) and are expressed as interaction patterns that can be used at runtime to monitor a worker's conformance with the patterns. The method has been coined by Rzeszotarski and Kittur (2011), demonstrating the effectiveness of the approach in predicting output quality. Kazai and Zitouni (2016) even conclude that "accuracy almost doubles in some tasks with the use of gold behavior data." As an extension, in Rzeszotarski and Kittur (2012) the authors propose CrowdScape, a system "that supports the human evaluation of complex crowd work through interactive visualization and mixed initiative machine learning."

4.3.4 Achievements. Verifying the achievement of predefined goals is a typical method used to assign so-called badges (certificates, proves of achievement) to users. The method resembles the merit badge concept of the Scout movement, where one must attain all preceding badges before qualifying for the next one (Puah et al. 2011). Achievements are used in crowdsourcing, especially in the context of gamified crowdsourcing tasks, such as exemplarily shown in Massung et al. (2013), where badges are used in a mobile data collection application to engage casual participants in pro-environmental data collection. Badges were earned for activities such as using the application for 5 days in a row or for rating a shop on the weekend.

4.3.5 Implicit Feedback. Implicit feedback is a method of content-based feedback analysis. Feedback is implicit and extracted from the behavior of evaluators, rather than from explicit feedback forms (De Alfaro et al. 2011). For example, in WikiTrust (Adler and De Alfaro 2007; Adler et al. 2011), a reputation management tool designed for assessing Wikipedia users, the reputation of the user is built based on the changes a user makes to the content. If a change is preserved by editors, the user gains reputation, otherwise he loses reputation. Lin et al. (2014) analyze implicit signals about task preferences (e.g., the types of tasks that have been available and displayed and the number of tasks workers have selected and completed) to recommend tasks to workers. Difallah et al. (2013) analyze workers' personal interests expressed in social networking sites (Facebook) to recommend tasks of likely interest in Mechanical Turk.

4.3.6 Association Analysis. Associations among people are, for instance, the friend-of relationships in Facebook, the recommendations in Freelancer, or the following relationship in Github. These relationships can be interpreted as expressions of trust or prestige and analyzed accordingly. Already in 1977, Freeman (1977) proposed the idea of betweenness centrality as a measure of a nodes importance inside a graph, with special attention toward communicative potential. In the specific context of crowdsourcing, Rajasekharan et al. (2013) have, for example, extended the well-known Page Rank algorithm (Page et al. 1999) with edge weights to compute what the authors call a "community activity rank" for workers to eventually assess their reputation. This kind of network analysis technique is typically more suited to contest crowdsourcing models, in which workers may know each other, and less to marketplaces where there is no communication among workers.

4.3.7 Task Execution Log Analysis. Given a log (trace) of worker interactions and task completions, it is possible to analyze and/or mine the log for assessment purposes. Fingerprinting uses log analysis to identify patterns; here the purpose is measuring quality attributes. Kucherbaev et al. (2016a), for example, use linear regression to estimate task duration times at runtime to identify likely abandoned tasks, that is, tasks that will not be completed by their workers. Moshfeghi et al. (2016) use game theory to classify workers based on task execution durations. Going beyond the estimation of individual quality attributes, Heymann and Garcia-Molina (2011) propose Turkalytics, a full-fledged analytics platform for human computation able to provide real-time insight into task properties like demographics of workers as well as location- and interaction-specific properties. Huynh et al. (2013) reconstruct from logs a provenance network that captures which worker saw/produced which data item in a set of inter-related tasks; the network allows the authors to predict the trustworthiness of task outputs. Jung et al. (2014) predict output quality by analyzing the time series of workers' past performance; KhudaBukhsh et al. (2014) do so to identify colluding workers.

4.3.8 Content Analysis. It is also possible to automatically analyze a task's description and text labels to assess properties like task difficulty or trust in the requester. Artz and Gil (2007), for instance, speak about "common sense" rules to make trust decisions, for example, do no trust prices below 50% of the average price. Difallah et al. (2013) propose the use of different methods to assess task difficulty: comparison of task description with worker skills, entity extraction and comparison (based on Linked Open Data), or content-oriented machine-learning algorithms. Also (Yang et al. 2016) come to the conclusion that "appearance and the language used in task description can accurately predict task complexity." Alagarai Sampath et al. (2014), instead, analyzed the semantic similarity of input field labels and showed that too diverse labels may act as distractors and that these can be used to predict accuracy.

4.3.9 Transfer Learning. Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned (Torrey

and Shavlik 2009). In crowdsourcing, transfer learning has been used to borrow knowledge from auxiliary historical tasks to improve the data veracity in a target task, for instance, to infer the gender or reliability of workers using a hierarchical Bayesian model (Mo et al. 2013). Fang et al. (2014) use transfer learning to estimate labelers' expertise in data labeling tasks inside Mechanical Turk. Zhao et al. (2013), instead, apply transfer learning in a cross-platform setting to transfer knowledge about workers from Yahoo! Answers to a Twitter-based crowdsourcing system.

4.3.10 Collusion Detection. Collusion detection aims to identify groups of colluding workers, that is, workers that share information to gain an advantage. KhudaBukhsh et al. (2014) aim to identify non-adversarial collusions in ratings by detecting strong inter-rater dependencies across tasks, as these diverge from the mean ratings. Marcus et al. (2012) inject ground truth data to detect colluders. Allahbakhsh et al. (2014), instead, compute a probability of collusion by analyzing past collaborations of workers on same tasks and recommend likely non-colluding workers for team formation.

5 QUALITY ASSURANCE

The logical step after assessing quality is assuring quality, that is, putting into place measures that help achieve quality—the more so if the assessment unveiled low quality for any of the attributes identified earlier. These measures, concretely, come in the form of strategies and actions that a requester may implement. Some of these strategies and actions are *reactive* if they react to clearly identified quality issues, for example, filtering outputs upon the verification that some outputs do not meet given quality thresholds. Other strategies and actions are instead *proactive* in that they do not need a triggering event to be applied, for example, following proper usability guidelines in the implementation of a task does not require a prior verification of usability.

Before looking into the strategies and actions that have been used so far for quality assurance in crowdsourcing, it is important to note that already assessing (measuring) quality, especially if the object of the assessment is people, may have positive side effects on quality. Most notably, reviewing has been shown to impact positively the performance of both workers and reviewers (Zhu et al. 2014) and quality in general (Hansen et al. 2013). Rating has been used to increase the requesters awareness of workers concerns and rights (Irani and Silberman 2013), but also rating the performance of workers has similar positive side effects (Dow et al. 2012). Many other studies that provide similar evidence exist.

In the following, we do not further study these side effects of assessment. Instead, we review the strategies and actions that specifically aim to improve quality as a first-order goal. Many of them require a prior quality assessment (e.g., filtering outputs requires first assigning quality labels to them), others do not. We explain these aspects next, and organize the discussion as illustrated in Figure 5. The identified *strategies* are as follows:

— *Improve data quality*: The first and most intuitive strategy to approach low quality in the output data is improving the quality of the data itself, where possible. Typical actions range from cleansing inputs (crowdsourcing is not immune to the garbage-in/garbage-out problem) to the iterative improvement of outputs.
— *Select people*: Another intuitive strategy is to identify workers that produce better results. Doing so may require requesters to filter workers by their profiles, recommending tasks to workers they think they will perform well or eliminating cheaters.
— *Incentivize people*: Incentivizing people means acting on the motivation that pushes people to perform well (or not). There are two different sub-strategies that aim to leverage on two different types of drivers (Rogstadius et al. 2011):

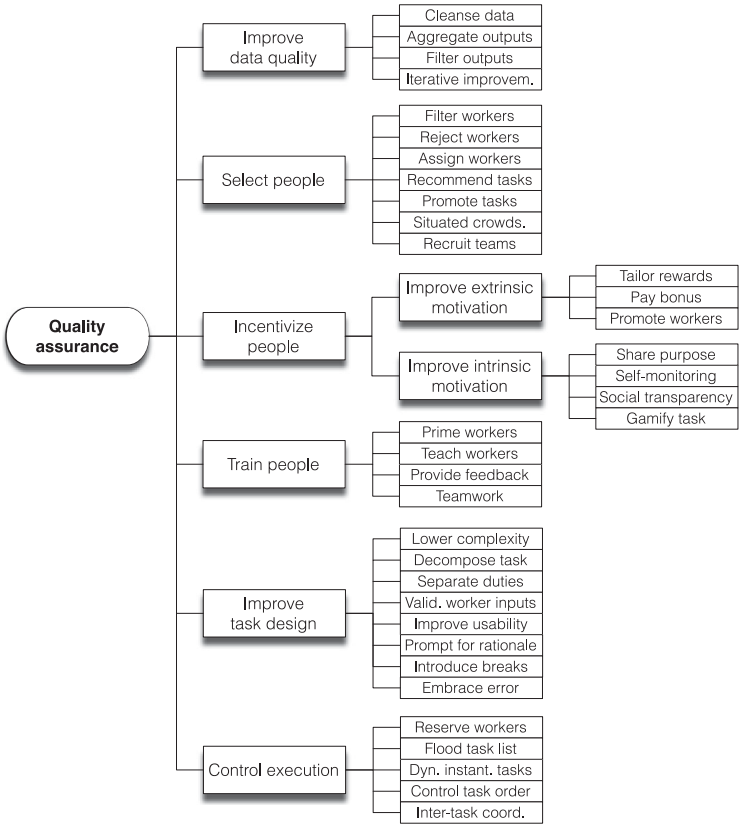

Fig. 5. The quality assurance *strategies* (boxes with shadow) and *actions* (boxes without shadow) emerged from literature.

- *Extrinsic motivation* depends on drivers that are determined from the outside of a person's own control. For example, workers may work harder if they see that it results into a higher reward or even a bonus from the requester.
- *Intrinsic motivation*, instead, depends on drivers that are internal to the person and does not depend on the desire for an external reward. For instance, workers may be pushed to perform better if they can compare their performance with that of other workers or if performing a task is entertaining (like in games).

— *Train people*: Workers can also be instructed or trained to be prepared better for specific tasks or domains and, hence, to perform better. Different approaches can be followed, such as teaching workers or providing feedback to the work they submit.

— *Improve task design*: One reason for low quality may be the low usability or understandability of the user interface workers are provided with or the task description and structure itself. Improving the design of a task may address related quality issues, for example, starting from input fields left empty.

— *Control execution*: Finally, some actions can be enacted during runtime, that is, during the execution of a task while workers are already working and submitting outputs. For example, if it is evident at some point in time that not all workers will produce outputs, it could be an idea to re-deploy some tasks.

Each of these strategies can be implemented using different, concrete *actions*. We discuss these in the following and summarize the discussion in Appendix A tabularly.

5.1 Improve Data Quality

5.1.1 Cleanse Data. The precondition of any process for good quality in output is good quality in input. Khazankin et al. (2012) point out that a crowdsourcing platform cannot be responsible for the output quality if the input data is inaccurate, and the requester is responsible for the accuracy of the inputs. In fact, workers may be reluctant to work on a task if they perceive the quality of the input (e.g., a blurred image) may impact their likelihood of getting the reward (Schulze et al. 2013). To overcome input quality problems, Bozzon et al. (2012), for instance, propose specific data pre-processing steps to assemble, re-shape, or filter input data in CrowdSearcher. Bigham et al. (2010) use computer vision techniques to improve the quality of pictures taken by blind people. Of course, data cleansing can be applied also to output data.

5.1.2 Aggregate Outputs. In the book "The Wisdom of Crowds," Surowiecki (2005) has shown with his ox weight guessing experiment that averaging (aggregating) the guesses of multiple people can produce an accurate overall guess. That is, adding redundancy (multiple workers working on a same task) and aggregating outputs can increase the quality of outputs, of course, at the cost of paying more workers. The responses of workers can also be weighted based on their reliability to increase the influence of responses given by trusted and skilled workers (Aydin et al. 2014). Particular attention is paid by literature to the aggregation of outputs in classification tasks (Ho et al. 2016; Gao et al. 2016; Ok et al. 2016; Liu and Wang 2012), including Boolean ones (De Alfaro et al. 2015; Ukkonen et al. 2015).

5.1.3 Filter Outputs. The assessment methods discussed in the previous section aim to tell apart "good" from "bad" items (outputs, workers, requesters, etc.). The corresponding assurance action that aims to improve quality is filtering out the bad items, so as to keep only the good ones. Filtering is very prominent in crowdsourcing. For instance, Dow et al. (2012) filter outputs based on self- and expert reviews, Hansen et al. (2013) filter outputs based on output agreements (with/without arbitration) and peer review, Rao et al. (2013) use majority votes to filter outputs, others use ground truth comparisons (Marcus et al. 2012), and so on. Jung and Lease (2012) filter workers based on their past performance. As an extension of the pure filtering of outputs, these may further be cleaned of possible biases by workers, for example, through active learning (Zhuang and Young 2015; Wauthier and Jordan 2011).

5.1.4 Iterative Improvement. Instead of asking one worker to evaluate the work of another worker (assessment), it is also possible to ask the former to directly improve the work of the latter (assurance). Multiple iterations of improvement are of course possible. Little et al. (2010a), for example, let workers iterate over writing tasks to incrementally improve a text written collaboratively by the crowd, while in Little et al. (2010b) they apply iterative improvement to decipher blurred text. In Turkomatic (Kulkarni et al. 2012a), workers can iteratively split tasks down until they are manageable, thereby reducing the complexity of the overall task.

5.2 Select People

5.2.1 Filter Workers. Similar to outputs, a requester may also filter workers to make sure they have the necessary skills, preparation, attitude, or similar for a task. This filtering can, for instance, be based on the worker profiles and look at skills and expertise (Allahbakhsh et al. 2013; Zhao et al. 2013), badges (CrowdFlower), or demographics (as a proxy for skills) (Kazai et al. 2012). Kazai et al. (2012) filter workers also by personality, while Li et al. (2014) look at worker reliability, and

Allahbakhsh et al. (2012) look at the reputation of workers inside the crowdsourcing platform. Hara et al. (2013) use statistical filtering (outlier elimination) to discard low-quality workers. Abraham et al. (2016), too, use reputation to filter workers, but also propose the use of adaptive stopping rules to lower the cost of tasks. Nushi et al. (2015) specifically filter workers to increase crowd diversity.

5.2.2 Reject Workers. Instead of selecting good workers, it is also possible to skim out (reject) non-fitting, fake, malicious, or colluding workers, often also called attackers or cheaters. Typical adversarial techniques are randomly posting answers, artificially generating answers, or duplicating them from other sources (Difallah et al. 2012) as well as collaborations among workers aimed at tricking the system (collusions). Bots or software agents can be eliminated using CAPTCHAs (Lasecki et al. 2014). Difallah et al. (2012) use start time and end time to predict if a given result comes from a cheater or not. It is important to properly tune the applied thresholds to detect cheaters without affecting good workers (Bozzon et al. 2013). Marcus et al. (2012) compare ground truth data with task outputs to prevent coordinated attacks from workers. Allahbakhsh et al. (2014) identify collusions from past collaborations of workers.

5.2.3 Assign Workers. Instead of waiting for tasks to be pulled by random workers, it may be more effective to proactively push tasks to cherry-picked workers. If both the skills required by a task and those possessed by workers are defined, tasks can be automatically assigned. For example, Mobileworks uses priority queues to assign tasks (Kulkarni et al. 2012b). Allahbakhsh et al. (2013) assign stronger workers to harder tasks, while Roy et al. (2015) specifically focus on knowledge-intensive tasks and show the benefits of maintaining a pre-computed index for efficient worker-to-task assignment. Difallah et al. (2013) show that matches can also be derived from social network activity, while Kulkarni et al. (2014) identify experts in social networks and recruit them. If no skills regarding tasks and workers are given, a two-phase exploration-exploitation assignment algorithm can be applied (Ho and Vaughan 2012). More dynamic approaches may even learn task assignment policies at runtime, for example, to maximize the expected information gain (Kobren et al. 2015).

5.2.4 Recommend Tasks. Instead of assigning tasks to workers, identified task-worker matches can also be used to provide workers with recommendations of tasks they very likely are interested in. This still allows workers to decide to work on a recommended task or not. A recommendation can be delivered, for example, as an email notification when a new task is posted on a platform (Bernstein et al. 2012), and subscriptions can be created using www.turkalert.com. Recommendations can be based on a worker's task browsing history (Yuen et al. 2015), or they may take into account implicit negative feedback (Lin et al. 2014).

5.2.5 Promote Tasks. With promoting a task we refer to all those actions that aim to enlarge the worker basis for tasks, even outside crowdsourcing platforms. More workers means more diversity, tasks completed, speed, and similar. Hu et al. (2012), for instance, place widgets with integrated tasks on third-party services, such as the International Children's Digital Library. Kulkarni et al. (2012b) ask workers to recruit others for a given task, while others incentivize this kind of recruitment (Nath et al. 2012). Also posting task information on worker community sites, such as the "HITsWorthTurkingFor" reddit page, can increase the task's workforce. A more direct way of promoting tasks is to invite people, for example, experts from online communities, to participate in tasks (Oosterman and Houben 2016).

5.2.6 Situated Crowdsourcing. This means bringing tasks to the workers physically, where it is more likely to encounter target workers, instead of waiting for them to join a crowdsourcing platform and to actively look for work. For instance, dedicated kiosks installed next to a library

entrance (Hosio et al. 2014), vending machines installed in front of the major lecture hall in a CS building (Heimerl et al. 2012), and public displays (Niforatos et al. 2016) have been used for the crowdsourcing of small tasks. Similarly, Vaish et al. (2014) ask mobile phone users for micro-contributions each time they unlock their phone, exploiting the common habit of turning to the mobile phone in spare moments.

5.2.7 Recruit Teams. Team-based recruitment approaches overcome the problem of recruiting enough workers by finding and recruiting teams of workers who have profiles matching the task requirements. Forming and recruiting teams is possible in smaller communities such as communities that are specialized in specific services like IT technical or business services (Vukovic et al. 2010; Schall et al. 2012). Retelny et al. (2014) identify experts in the crowd and organize them into teams to solve complex tasks. Li et al. (2014) run trial tasks to discover groups of workers producing higher quality than average and target them for future work on the same task. Rokicki et al. (2015) study different team competition designs to improve performance.

5.3 Improve Extrinsic Motivation

5.3.1 Tailor Rewards. One of the key properties of each task is the reward. Choosing the right form and/or amount of the reward is thus of utmost importance for the success of a crowdsourced task. Faradani et al. (2011) and Ho et al. (2015) show that it is important to tweak the amount of the reward properly, so as to obtain good results. Radanovic and Faltings (2016) demonstrate the effectiveness of dynamically adjusting payments based on output quality. Mao et al. (2013) study the effectiveness of different rewarding schemas, such as volunteering, pay per time, pay per task, pay per each data unit in a task, and show that workers are sensitive to the schemas. Along the same line, Ikeda and Bernstein (2016) show that paying tasks in bulks may increase the task completion rate, while coupons or material goods decrease participation. Scekic et al. (2013a) also discuss deferred compensation and relative evaluation as well as team-based compensation schemas. Sakurai et al. (2013) propose reward strategies based on worker performance. Singer and Mittal (2013) study how to maximize the amount of tasks completed with a given budget using different pricing strategies. Rokicki et al. (2014) study gambling-based rewarding strategies. Of course, also non-monetary rewards, such as badges, can be seen as "virtual trophies" (Scekic et al. 2013a) that motivate workers, yet Kobren et al. (2015) also show that these kinds of objectives must be designed carefully, otherwise they may produce detrimental effects.

5.3.2 Pay Bonus. A bonus is an additional, discretionary reward for good performance added to the base reward of a task. Bonuses are typically granted for the achievement of predefined goals or for reaching the threshold of key performance indicators (KPIs) (Scekic et al. 2013a). Difallah et al. (2014) grant bonuses in response to workers meeting given milestones. Yin et al. (2014) top-up the base reward for tasks if workers provide correct answers and if they react in less than 1 second. Yu et al. (2014) give credits as base reward, bonuses of 5 cents for additional tasks performed and of 10 US dollars for earning most credits (assigned every other month). Faltings et al. (2014) show that game-theoretic bonus schemas can also help eliminate worker bias, next to incentivizing them to work better.

5.3.3 Promote Workers. Promoting a worker means raising the worker to a higher position compared to his/her current one or compared to others. Promotions are particularly suitable to those environments that are characterized by a long-lasting engagement of workers, for example, crowdsourcing environment with deferred payment schemas. It has been shown that the prospect of a promotion, for example, to get higher rewards or to obtain access to new types of tasks, increases motivation, also over longer periods of time (Scekic et al. 2013a). For example, Dow et al.

(2012) promote workers from content producers to feedback providers (assessors), while Scekic et al. (2013b), next to promotions, also introduce the idea of punishments.

5.4 Improve Intrinsic Motivation

5.4.1 Share Purpose. Tasks that have a purpose that goes beyond the individual micro-task, that workers understand and can identify with can help attract crowds to perform tasks with higher motivation, also for free by volunteering (Dontcheva et al. 2014). Examples of crowdsourcing initiatives driven by this sense of purpose are Zoouniverse (www.zooniverse.org) described in Mao et al. (2013) or Wikipedia. As workers contribute for the purpose, rather than for the monetary reward (if any), these tasks are typically less attractive to spammers or adversarial workers. Kobayashi et al. (2015) show that tasks with an explicit social purpose may help attract especially senior citizens as workers. Kaufman et al. (2016) identified that people tend to contribute more to volunteering tasks with few other contributors involved.

5.4.2 Self-Monitoring. Enabling workers to compare their performance with that of other workers (self-assessment) can switch workers into a competition mode that pushes them to perform better (Ipeirotis and Gabrilovich 2014; Scekic et al. 2013a). Most crowdsourcing platforms today already assign workers a reliability or performance rating visible in the platform. Ipeirotis and Gabrilovich (2014) study the benefits of displaying scores to workers individually, overall crowd performance, and leaderboards with complete or partial rankings of workers. Leaderboards have been used extensively so far (Rokicki et al. 2014; Dontcheva et al. 2014; Preist et al. 2014).

5.4.3 Social Transparency. Social transparency means sharing identity (e.g., real name, persistent identity in applications), content (e.g., comments), actions (e.g., tasks performed, endorsements) among workers (Huang and Fu 2013a). Through transparency, workers build trust and bond with their co-workers and the requester and define standards and quality control mechanisms that eventually may improve performance and output quality (Huang and Fu 2013a; Viégas et al. 2007). According to Yu et al. (2014), maintaining good relationships between workers helps also to attract more workers. In collaborative crowdsourcing scenarios, social pressure among workers (e.g., asking to stay) can have positive impacts on performance (Feyisetan and Simperl 2016).

5.4.4 Gamify Task. Luis von Ahn introduced Games With A Purpose, where participants perform tasks in games for joy and entertainment, rather than financial reward (Ahn 2006). Krause and Kizilcec (2015) found that for complex tasks workers produce better results in a gamified than in a paid condition; for simple tasks there was no difference. Designing tasks that induce curiosity is a task-agnostic strategy to improve worker retention (Law et al. 2016). Feyisetan et al. (2015) propose a model to predict optimal combinations of financial and gamified incentives.

5.5 Train People

5.5.1 Prime Workers. Priming uses implicit mechanisms to induce observable changes in behavior that are usually outside of the conscious awareness of the primed person (Morris et al. 2012). Images, text, music, and videos can be used to induce positive emotions in workers that in turn result in a positive effect on task performance, for example, an image of a laughing child may induce workers to perform better (Morris et al. 2012). Alagarai Sampath et al. (2014) use priming to help workers remember information. Faltings et al. (2014) study the effectiveness of priming on task performance.

5.5.2 Teach Workers. Teaching means providing workers with suitable instructions in order to enable them to perform tasks. Doroudi et al. (2016) show that providing expert examples and letting workers validate others' contributions are effective ways of teaching. Singla et al.

(2014) use an algorithm to select expert examples to show to users. To many workers, gaining or improving skills is a motivation per se. This motivation can be enforced by in-person training like in Samasource (http://www.samasource.org/) or Mobileworks (https://www.mobileworks.com/) or through interactive tutorials, such as the ones described by Dontcheva et al. (2014). Also, designing tasks in a way that helps workers learn improves output quality (Yu et al. 2014).

5.5.3 Provide Feedback. Workers getting feedback from requesters about their performance provide results of better quality (Dow et al. 2012). The process of reviewing others' work itself improves quality (Yu et al. 2014) and helps workers gain new skills (Zhu et al. 2014). Kulkarni et al. (2012a) show how requester feedback is important to handle complex work in their Turkomatic platform.

5.5.4 Team Work. Team work means working together on a same task, where "together" means by interacting with each other (we do not consider it a team work if several workers jointly label a set of images without any inter-personal communication). Kittur (2010), for instance, use team work for the translation of a poem, which requires negotiation between workers. André et al. (2014) provide workers with a shared, collaborative editing space for creative writing tasks. Dorn et al. (2012) propose flexible workflows to organize teams of collaborating workers.

5.6 Improve Task Design

5.6.1 Lower Complexity. One of the challenges in crowdsourcing is identifying the right granularity, that is, complexity, for tasks. As Rogstadius et al. (2011) show, the accuracy of outputs typically decreases with increasing task complexity. From a design perspective, it is further important to implement tasks in a way that limits cognitive complexity; for instance, comparing two objects is easier than identifying features of individual objects (Anderton et al. 2013). While the simplification of more complex tasks introduces longer completion times, it leads to higher quality; simpler tasks suit better if workers perform with interruptions (Cheng et al. 2015b).

5.6.2 Decompose Task. Another way to lower the complexity of a task is to decompose it into sub-tasks. Kittur et al. (2011) propose a partition-map-reduce approach in CrowdForge, where work is split into a set of sub-tasks executed in parallel and merged back later. Kulkarni et al. (2012a) propose a price-divide-solve algorithm in Turkomatic, where workers themselves can decide whether to work on a task themselves or rather split it into sub-tasks and then merge the outputs produced by others.

5.6.3 Separate Duties. Separation of duties is a design pattern lent from the business practice that requires multiple people to be involved in a decision, so as to prevent fraud or errors. Organizing work in such a way that workers have only one single task to perform may further lead to higher quality of outputs. Bernstein et al. (2010), for instance, propose a find-fix-verify approach for text proofreading, where some workers identify errors in a text (find), some others correct identified errors (fix), and others again check if there are no further mistakes left (verify). In Kulkarni et al. (2012a), workers decide if they prefer to split a task or to perform it.

5.6.4 Validate Worker Inputs. Validating worker inputs means checking that the values provided by the workers through the form fields in the task UI comply with formal requirements, for example, no empty fields or only correct email addresses. This is a common design guideline that is, however, often not followed by requesters, eventually leading to low output quality. In Deluge (Bragg et al. 2013) tasks are designed in such a way that workers must select at least one item from a list of available options. On CrowdFlower and AskSheet (Quinn and Bederson 2014), the

requester can select allowed formats (e.g., an email, US address, phone number, or even a custom regular expression) for fields in the task form and identify mandatory and optional input fields.

5.6.5 Improve Usability. Just like for any kind of software, it is important that task UIs are usable and properly follow common usability guidelines (Nielsen et al. 2002; Khanna et al. 2010). Well-designed tasks lead to higher quality of the outputs (Kazai et al. 2011a). For instance, it is a good practice to provide workers with clear and meaningful examples of good work (Willett et al. 2012). Highlighting input fields adequately and placing them closer to the relevant content reduces search time and working memory load (Alagarai Sampath et al. 2014). Designing tasks in a way that it takes the same or less time to perform a task properly rather than to cheat also helps avoid spammers (Kittur et al. 2008).

5.6.6 Prompt for Rationale. Collecting rationales from workers for their own work is a good way to encourage workers to be more conscious and to collect verifiable results, especially for subjective tasks (McDonnell et al. 2016). Drapeau et al. (2016) suggest that allowing workers to adjust their work based on rationales provided by other workers may improve quality further.

5.6.7 Introduce Breaks. Performing long sequences of monotonous tasks can be downing for workers. Dai et al. (2015) show that introducing occasional breaks, such as playing games or reading a comic, may help increase workers' retention.

5.6.8 Embrace Error. There are classes of tasks where fast completion has higher priority than the quality of each individual worker's judgment. A way to approach such tasks is to design them encouraging workers to perform very fast, accepting and even embracing possible errors, which can be later rectified through suitable post-processing (Krishna et al. 2016).

5.7 Control Execution

5.7.1 Reserve Workers. Maintaining a pool of workers ready to work on a task is an effective approach to minimize waiting time that is especially useful if results are to be collected fast, for example, in real-time. For instance, it is possible to pay workers for the time they spend waiting for tasks (Bernstein et al. 2012). To make sure workers stay focused during the waiting time they can be asked to play a game (Lasecki et al. 2013). In case of requesters launching tasks only occasionally, it can be financially efficient to maintain a single pool of workers for multiple requesters. Bigham et al. (2010) adopt this approach and notify workers of the retainment pool when a new task is published.

5.7.2 Flood Task List. Chilton et al. (2010) have shown that workers tend to give more attention to newer tasks when looking for tasks to work for. If a task does not appear on the top of the first page of the task list or even goes to the second page of the listing, the attention devoted to it drops dramatically. To keep the attention of workers high, Bernstein et al. (2012) have shown that repeatedly posting tasks (flooding) inside a crowdsourcing platform indeed increases the task's visibility and attractiveness.

5.7.3 Dynamically Instantiate Tasks. Monitoring the work of the crowd can allow the requester to identify quality issues while a task is still in execution, for example, workers not completing their work or doing so with too low level of quality. Kucherbaev et al. (2016a) show how dynamic re-launching of tasks, that are automatically identified as abandoned, helps to lower overall task execution times in exchange of a small cost overhead. Bozzon et al. (2013) support the dynamic re-planning of task deployments. Yan et al. (2011) study how to actively learn to choose workers to minimize speed and/or time. Many study how to maximize quality by dynamically instantiating and assigning work under budget constraints (Li et al. 2016; Tran-Thanh et al. 2015; Chen et al.

2013; Karger et al. 2011, 2014). Bansal et al. (2016) use content similarity to dynamically identify data items to label and propagate labels to similar items.

5.7.4 Control Task Order. If a group of tasks presents interdependencies where the output of one task affects the usefulness of another task, controlling the order of task deployment can help avoid useless extra costs. For instance, in a set of comparison tasks if $a = b$ and $b \neq c$, then there is no reason to compare also a and c (Vesdapunt et al. 2014). Marcus et al. (2012) show how to use selectivity estimations used in traditional databases for query optimization to order tasks and reduce them in number. Lasecki et al. (2015) and Newell and Ruths (2016) suggest to group tasks with related content into batches, because tasks that have already been completed by a worker affect his/her focus in subsequent tasks, and the too high a diversity of task content inside a same batch of tasks leads to interruptions due to context switch. Yet, Eickhoff and de Vries (2013) also show that large batches tend to attract more cheaters than small batches. Difallah et al. (2016) address context switch and minimize latency using scheduling techniques guaranteeing that all tasks get equal attention.

5.7.5 Inter-Task Coordination. More complex tasks, especially composite tasks that involve multiple different sub-tasks, can be managed by automating the respective crowdsourcing workflow. Bozzon et al. (2013) propose an event-condition-action approach to organize tasks. Kucherbaev et al. (2016b) overview workflow automation instruments tailored to crowdsourcing, for example, Turkit (Little et al. 2010a) supporting scripting and CrowdLang (Minder and Bernstein 2012) supporting visual modeling.

6 ANALYSIS OF STATE-OF-THE-ART CROWDSOURCING PLATFORMS

In the following, we discuss and compare a selection of state-of-the-art crowdsourcing platforms with the help of the taxonomy introduced in this article. We thus specifically look at the quality model/attributes, the assessment methods, and the assurance actions supported by the approaches.

The crowdsourcing platforms we analyze represent a selection of heterogeneous instruments drawn from both industrial systems and academic research prototypes. The selection is by no means intended to be complete, nor does it represent a list of "most popular" instruments. It is rather the result of an internal discussion among the authors of this article of the platforms we found in our research, the platforms we looked at in the analysis, and the platforms we personally worked with over the last years. The goal of the selection was to provide a varied picture of the platforms that characterize today's crowdsourcing landscape.

The result of this discussion is the selection of the following 14 platforms: *Mechanical Turk* (http://www.mturk.com), one of the first crowdsourcing platforms for paid micro-tasks; *CrowdFlower* (http://www.crowdflower.com), a meta-platform for micro-tasks that acts as proxy toward other platforms; *MobileWorks* (http://www.mobileworks.com), a platform with an ethical mission that pays workers hourly wages; *Crowdcrafting* (http://crowdcrafting.org), which targets scientists and non-paid volunteers; *Turkit* (Little et al. 2010c), a JavaScript-based language for the programmatic coordination and deployment of tasks on Mechanical Turk; *Jabberwocky* (Ahmad et al. 2011), a MapReduce-based human computation framework with its own programming language; *CrowdWeaver* (Kittur et al. 2012), a model-based tool with a proprietary notation and crowdsourcing-specific constructs; *AskSheet* (Quinn and Bederson 2014), a Google Spreadsheet extension with functions for the integration of crowdsourcing tasks; *Turkomatic* (Kulkarni et al. 2012a), a crowdsourcing tool that delegates not only work to the crowd but also task management operations (e.g., splitting tasks); *CrowdForge* (Kittur et al. 2011), a crowdsourcing framework similar to Turkomatic that follows the Partition-Map-Reduce approach; *Upwork* (https://www.upwork.com/), an auction-based platform for freelancers in different domains (e.g.,

Table 1. Crowdsourcing Platforms Comparison by Supported Quality Model/Attributes, Assessment Methods, and Assurance Actions

	Quality attributes	Assessment	Assurance
MTurk	data (accuracy), task incentives (extrinsic), task terms and conditions (privacy), task performance (cost efficiency, time efficiency), worker profile (location), worker credentials (skills), worker experience (badges, reliability)	rating (reliability), qualification tests (skills), ground truth questions (accuracy), rating-based achievements (badges)	filter workers by reliability (acceptance rate, tasks completed), by badges (domain-specific master skill) or location; assign workers; reject workers; tailor rewards; pay bonus; provide feedback to workers; validate worker inputs; prompt for rationale
CrowdFlower	data (accuracy, timeliness), task description (clarity, complexity), task incentives (extrinsic), task terms and conditions (privacy, information security), task performance (cost efficiency, time efficiency), worker profile (location), worker credentials (skills), worker experience (badges, reliability)	rating (reliability), qualification tests (skills), output agreement (accuracy, reliability), feedback aggregation (task satisfaction survey on clarity, complexity, incentives), ground truth questions (accuracy), rating-based achievements (badges), execution log analysis (accuracy, timeliness)	aggregate outputs (consistency, accuracy), filter workers (by country, by distribution channel, by NDA), badges (obtainable level), skills (language), reject workers, tailor rewards; pay bonus, provide feedback, validate worker inputs, prompt for rationale, control task order (select an order in which tasks are deployed), inter-task coordination (Crowdflower workflow plugin)
MobileWorks	data (accuracy, timeliness), task incentives (extrinsic), task terms and conditions (privacy), worker profile (age, gender, location), worker credentials (skills), worker experiences (badges, reliability), group (availability)	rating (reliability), qualification tests (skills), expert review (accuracy and reliability of group members), peer review (accuracy), ground truth (accuracy)	filter workers, reject workers, assign workers, promote tasks (workers involve referrals), recruit teams (recruit local small teams, interview via Skype), tailor rewards, pay bonus, promote workers to team leaders, teach workers, provide feedback, teamwork, validate worker inputs, prompt for rationale, control task order
CrowdCrafting	intrinsic incentives (citizenscience), task terms and conditions (privacy), task performance (time efficiency), worker profile (location, personal details)	execution log analysis (time efficiency)	promote tasks (featuring on the platform), share purpose (tasks from high impact scientific fields), self-monitoring (contributions leaderboard), social transparency (optional public worker profiles), prompt for rationale, control task order (tasks order priority, scheduling - depth first, breadth first, random)
TurKit	data (accuracy, timeliness), extrinsic incentives (reward)	voting (accuracy)	filter outputs, iterative improvement, tailor rewards, separate duties (explicit voting tasks), dyn. instantiate tasks (as a part of iterative improvement), control task order (via automatic workflow), inter-task coordination (programmatically using JavaScript-like scripts)
Jabberwocky	data (accuracy), extrinsic incentives (reward), worker profile (age, gender, location, custom attributes), credentials (skills, certificates)	rating (accuracy), voting (accuracy)	aggregate outputs, filter outputs, iterative improvement, filter workers (rich profiles), assign workers, tailor rewards, inter-task coordination (Dog programs)
CrowdWeaver	data (accuracy, timeliness), extrinsic incentives (reward), task performance (cost efficiency, time efficiency)	voting (accuracy), output agreement (accuracy), ground truth (accuracy)	cleanse data (divide, permute tasks), inter-task coordination (supports runtime workflow edits)
AskSheet	data (accuracy), extrinsic incentives (reward), task performance (cost efficiency)	rating (accuracy), voting (accuracy)	cleanse data (facilitated through spreadsheet paradigm), aggregate outputs (spreadsheet formulas), filter outputs (spreadsheet formulas, worker vote), tailor rewards, validate worker inputs (enforce predefined bounds and types), dyn. instantiate tasks (launching extra instances until a certain threshold is reached), control task order (prioritization), inter-task coordination (using referral links in formulas)
Turkomatic	data (accuracy), extrinsic incentives (reward), task description (complexity)	voting (accuracy)	aggregate outputs (via "merging" tasks), filter outputs (via "voting" tasks), tailor rewards, teamwork (via runtime workflow edits), decompose task (via "subdivision" tasks), inter-task coordination (emerges at runtime following the price-divide-solve algorithm)
CrowdForge	data (accuracy), extrinsic incentives (reward), task description (complexity)	voting (accuracy)	aggregate outputs (via "reduce" step), filter outputs (via "voting" tasks), decompose task (via dynamic partitioning), inter-task coordination (according to partition-map-reduce approach with possible nesting)

(Continued)

Table 1. Continued

	Quality attributes	Assessment	Assurance
Upwork	data (accuracy), extrinsic incentives (reward), task terms and conditions (IP), task performance (cost efficiency), requester (reputation), worker profile (location), worker credentials (skills, certificates, portfolio), experiences (badges, reliability)	rating (accuracy), qualification test (skills), referrals (reliability), expert review (accuracy), achievements (badges), execution log analysis (monitoring of worked hours)	filter workers (also via interviews), reject workers (disputes), assign workers (invite to work), recruit teams, tailor rewards (per hour vs. fixed price), pay bonus, social transparency, provide feedback, teamwork, prompt for rationale
99designs	data (accuracy, timeliness), extrinsic incentives (reward), task terms and conditions (IP), worker profile (location), worker credentials (skills), worker experiences (badges, reliability)	rating (accuracy, reliability), referrals (reliability), achievements (badges)	filter outputs (based on competition), filter workers (profile, experience), assign workers (invite to work), tailor rewards (predefined plans or direct negotiation), self-monitoring (submissions of others are optionally visible), social transparency (workers can use real identities and build professional profiles), provide feedback, teamwork, prompt for rationale
Topcoder	data (accuracy, timeliness), extrinsic incentives (reward), task terms and conditions (IP, information security), worker profile (location), worker credentials (skills), experiences (badges, reliability, custom performance metrics)	rating (accuracy), peer review (called community review), achievements (badges), content analysis (unit tests on submitted code)	filter outputs (test-based or community review), tailor rewards, pay bonus, self-monitoring (leaderboard), social transparency (workers can link social network profiles with their identities), prompt for rationale
Innocentive	data (accuracy), extrinsic incentives (reward), task terms and conditions (IP, compliance), worker profile (location, rich profile), worker credentials (skills, certificates)	rating (accuracy), expert review (accuracy)	filter outputs (proposals filtered manually), recommend tasks (based on workers skills and interests), tailor rewards, prompt for rationale

software development or writing); *99designs* (http://99designs.it/), a contest-based platform for graphical design freelancers; *Topcoder* (https://www.topcoder.com/), a contest-based platform for software developers and designers; *Innocentive* (http://www.innocentive.com/), a platform for the crowdsourcing of innovation challenges.

We use the taxonomy to classify each of these platforms or tools individually in Table 1 and to summarize the supported quality attributes, assessment methods, and assurance actions from a qualitative point of view. Figures 6–8 provide a quantitative summary of the table in the form of heat maps that color each of the leaves (quality attributes, assessment methods, and assurance actions, respectively) of Figures 3–5 with a different intensity according to how many of the platforms/tools support the respective feature; possible values thus range from 0 (white) to 14 (dark green). We discuss the findings in the following Sections.

6.1 Quality Model

Figure 6 illustrates how many of the studied platforms support each of the attributes of the quality model derived in this article. Immediately, it is evident that the core concern almost all platforms (13 out of 14) have is the accuracy of the outputs produced by the crowd. This is not surprising, as high-quality outputs are one of the key drivers of crowdsourcing in the first place (next to cost and time). In order to allow the requester to tweak quality, most platforms allow the requester to fine-tune the extrinsic incentives (rewards) given for tasks (13/14), to select workers based on age (2/14), gender (2/14), location (9/14), and skills (8/14). In addition, approximately half of the platforms also implement proper reliability tracking (6/14) or reputation management systems (6/14) for worker selection.

In order to understand these numbers better, it is necessary to disaggregate them. For instance, it is important to note that worker profiles are typically kept simple by the marketplace platforms (e.g., Mechanical Turk or CrowdFlower), while they are more sophisticated for auction-/contest-

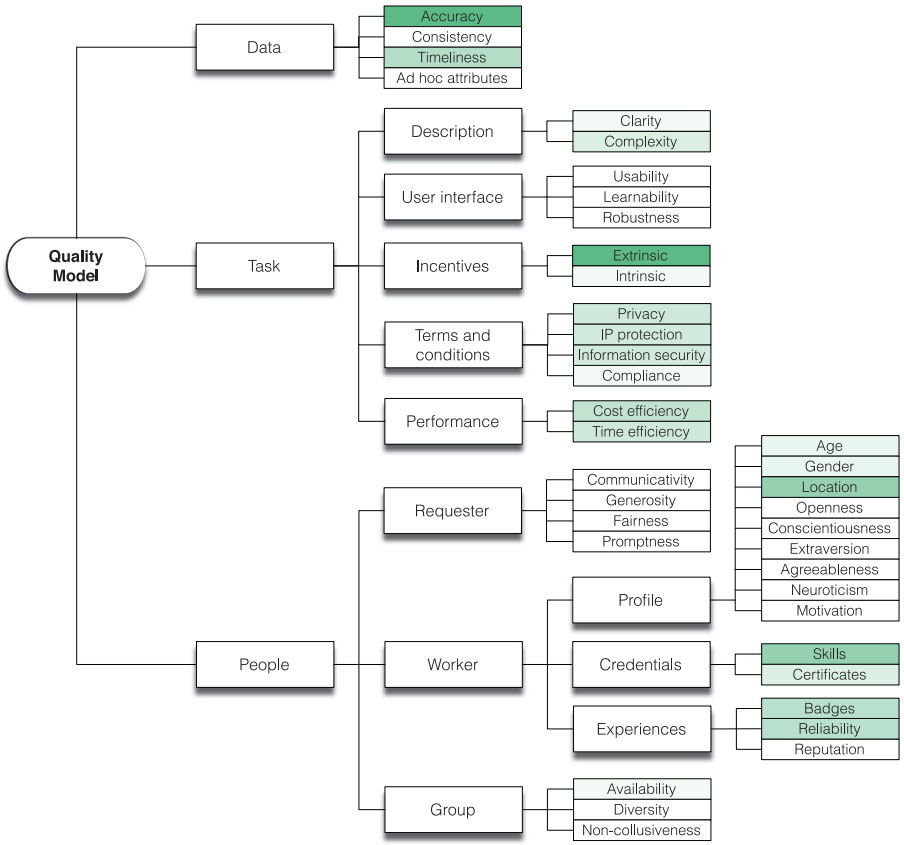

Fig. 6. Heat map of the quality model.

based platforms that target skilled freelancers (e.g., Topcoder or 99designs). In fact, the former (but also the research prototypes that target the coordination of multiple tasks on top of existing marketplace platforms) mostly focus on simple microtasks for which the problem usually is finding workers at all, not finding the best ones possible; in the latter platforms, instead, the profile has an advertising purpose as well and therefore plays a more important role. All platforms generally protect the privacy of their users; the research prototypes adopt the policies of the underlying platform, while the platforms for freelancers may disclose personal information to enable transactions among collaborating actors. Support for IP protection comes in the form of NDAs or transfer agreements for freelancers. Worker assessment seems mostly based on skills, reliability, badges, and/or reputation. That is, crowdsourcing tends to be meritocratic.

However, overall there is only little support for the different quality attributes identified in these articles. Mostly, this is explained by the different focus of research (wide spectrum of attributes) and commercial practice (narrow focus). For both areas, we identify the following points as possible future research directions:

— *Personality.* The character and behavior of workers and requesters, acknowledged by research as directly impacting the quality of outputs and the satisfaction of both, is commonly neglected by state-of-the-art instruments. Yet, there seems to be an opportunity for platforms that also aim to improve the attitude of people, for example, by facilitating

the creation of shared values, social contacts, or social norms. People that feel well in their workplace perform better and are more engaged in their work.
— *Transparency.* In general, while workers on many platforms are anonymous to requesters, it is important to note that requesters are even more anonymous to workers. In order to increase mutual trust, it is advisable that also requesters be assessed properly and participate more actively in the actual work. How this assessment and/or collaboration could happen is not straightforward in a domain where each new task may put into communication completely new and unknown actors.
— *Group work.* While there are first attempts of organizing workers into groups and to facilitate the collaboration between workers and requesters, the quality and benefit of group work is still not fully studied and understood. In this respect, we believe the attributes considered so far are not sufficient to help characterize the quality of and leverage on the full power of the crowd. But first of all, more support for group work and collaboration by the platforms themselves is needed.
— *User interface quality.* Surprisingly, only very little attention is paid by the studied platforms to the quality of the user interface of tasks deployed by requesters. Some platforms (e.g., CrowdFlower) provide crowdsourcing as a service to their customers, whereby they also design and control the quality of the respective UIs. Yet, on the one hand, usability and learnability are still concepts that have not percolated into concrete tools and, on the other hand, they are still too generic as attributes to really help requesters to develop better interfaces.

6.2 Quality Assessment

Figure 7 illustrates the support of the discussed assessment methods. As in the case of the quality model, also here we see that only about half of the methods identified previously are also implemented by current crowdsourcing platforms. Rating (9 out of 14) and voting (6/14) are the most prominent assessment methods, the former mostly used by requesters to assess work/workers, the latter mostly used by workers for the peer assessment of work/workers. It is further evident that the current support of assessment methods is mostly limited to technically simple methods, while more complex capabilities are crowdsourced themselves. For instance, the group assessment methods are well developed overall, while the computation-based methods are still limited.

Again, it is good to disaggregate the numbers. While rating is almost the only feedback mechanism in the marketplace platforms, and it is essential for the proper functioning of the platforms based on contests, it is interesting to note that almost none of the research prototypes makes use of rating. Instead, given their focus on the coordination of tasks, these platforms heavily leverage on voting for quality assessment, an activity that naturally involves multiple workers and possibly the requester and external experts. In line with the limited user profiles featured by typical marketplace platforms, they instead prominently propose the use of qualification tests to select workers. On the contrary, the auction- and contest-based platforms bet on achievements as an automated technique to assess performance (e.g., badges).

For the future, we identify the following challenges in terms of quality assessment:

— *Self-assessment.* This assessment method is underestimated by current practice, despite its proven benefit to learning and to the quality of produced outputs. How to make self-assessment an integral and integrated part of crowdsourcing in general is, however, non-trivial and still open.
— *User interface assessment.* In line with our comment on the quality attributes regarding UI quality, also in terms of assessment methods there is huge room for improvement. Significant effort still needs to be invested into the development of proper guidelines for

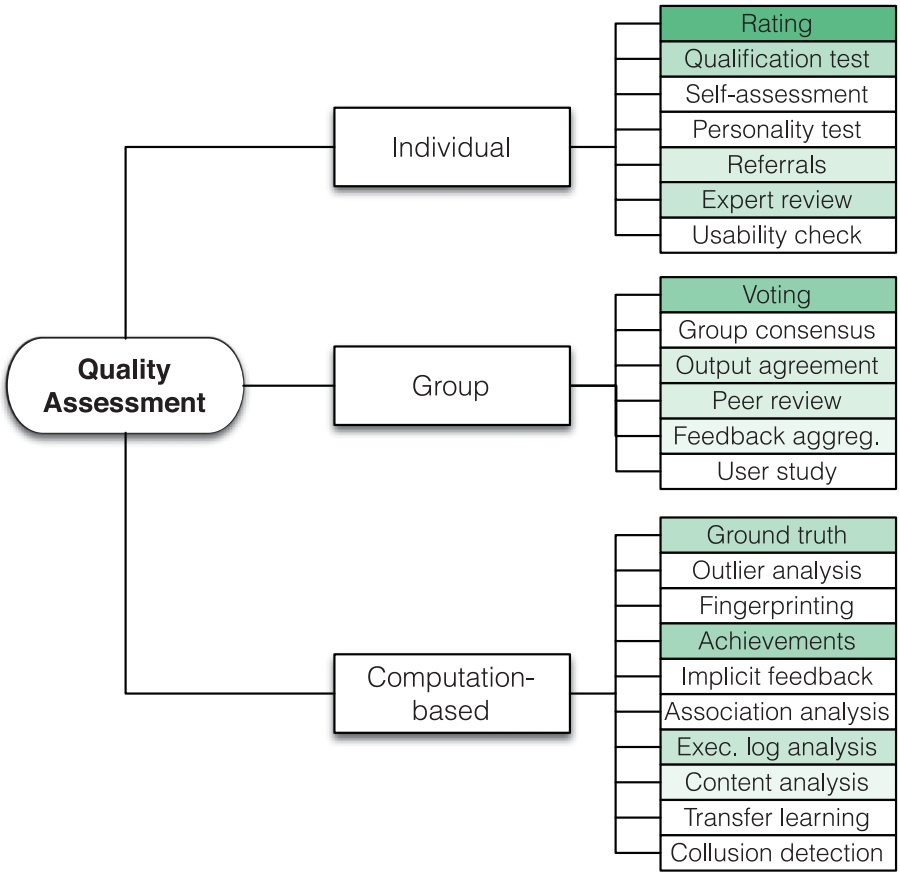

Fig. 7. Assessment heat map.

the design of intuitive and robust task UIs, as well as into automated methods of their assessment, for example, as attempted by Miniukovich and De Angeli (2015).

— *Runtime analytics.* Assessment methods are mostly applied after task execution, while they could easily be applied also during task execution to enable preemptive interventions aimed at increasing quality while a task is being worked on. Suitable analytics features and interventions could not only improve the accuracy of outputs, but also their timeliness and cost efficiency.

6.3 Quality Assurance

Finally, Figure 8 illustrates the state of the art in quality assurance. It is surprising to see that the spectrum of assurance actions introduced earlier in this article is explored almost completely by current crowdsourcing platforms. Of course, the tailoring of the reward (11 out of 14) dominates, but also filter outputs (8/14) and filter workers (6/14) are adopted relatively widely. Prompting for rationals is supported where requesters can design task input forms or interact with workers (8/14). Only situated crowdsourcing, prime workers, improve usability (out of the control of platforms), reserve workers, and flood task list are not supported by any of the platforms.

The disaggregation of the data reveals that the strong support of inter-task coordination (7/14) and task order control (5/14) mostly stems from the research prototypes that specifically focus on

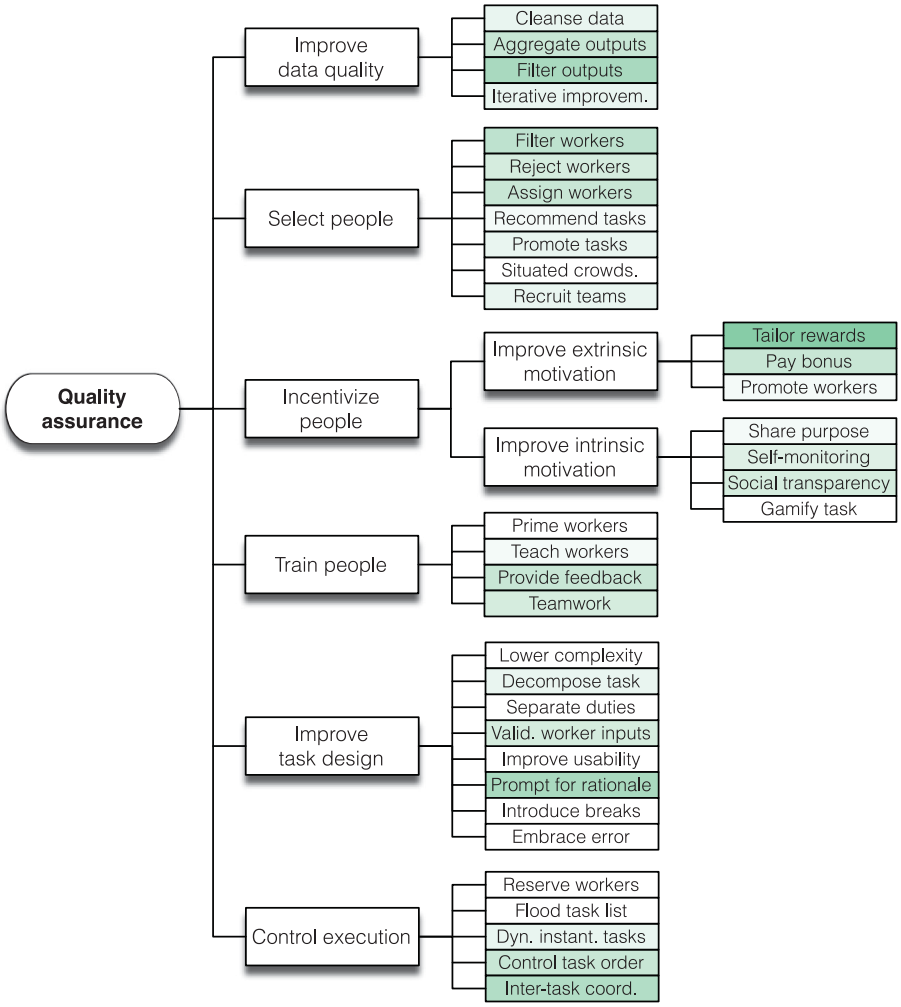

Fig. 8. Heat map of assurance model.

this aspect. To the best of our knowledge, only CrowdFlower internally uses a workflow engine for task automation, and only Innocentive seems to make use of task recommendations. Interestingly, it appears that the research community with its prototypes for task automation mostly concentrates on assuring quality by promoting actions with effects that are limited to a given context only, for example, controlling the order of tasks or splitting tasks into smaller chunks. Commercial platforms rather look at more comprehensive actions, such as intrinsic motivators (e.g., social transparency) or team building, which have effects that cross tasks and are longer lasting in nature.

We identify the following points as crucial aspects to approach next:

— *Task recommendation.* With the increase of the popularity of crowdsourcing, both the number of workers and that of tasks published on crowdsourcing platforms is destined to grow. It is thus increasingly hard for workers to find tasks they are interested in, capable of, and good at. This asks crowdsourcing platforms to help match workers and work, which means

recommending tasks to workers with the right skills. Research is very active in this area, but support for custom worker profiles and recommendation algorithms is still missing.
— *Long-term relationships.* To make crowdsourcing sustainable in the longer term, it may be necessary that requesters and workers establish tighter relationships, for example, to train workers with skills that will serve them in the future and assure them predefined levels of income. Continuous and task-specific training must turn into common practice and be seen as an investment by both workers and requesters.
— *Workflow integration.* Finally, with the increasing complexity of work that is crowdsourced, the need for coordination and automation will increase too. So far, the problem has been studied only in the form of isolated research prototypes. The challenge now is conceiving principles, techniques, and tools that enable the seamless integration of crowd workflows into existing IT and business practices.

7 CONCLUSION AND OUTLOOK

By now, crowdsourcing is a well established practice and a concrete option to solve problems that neither individuals nor computers may be able to solve on their own (nor together), while they can be solved by asking help from contributors that are not known but that can bring in their human intelligence. With this survey, we comprehensively studied one of the key challenges of crowdsourcing, that is, quality control. We analyzed literature on crowdsourcing published in major journals and conferences from 2009 onward and synthesized a quality model that represents all the attributes that have been studied so far to understand quality in crowdsourcing. We accompanied the model with a comprehensive discussion of the methods and actions that have been used to assess quality and to enforce quality, respectively. From the survey it is evident that, although quality control is perceived as crucial by all actors involved in crowdsourcing and significant effort has already been invested into it, we are still far from a practice without quality issues that effectively delivers human intelligence to its customers.

We consider the following two areas for future work as particularly critical to guarantee quality and sustainability in the longer term:

— *Domain-specific services.* Most crowdsourcing platforms, especially micro-task platforms and research prototypes, still position themselves as technology providers managing the crowd and tasks from an abstract point of view. Crowdsourcing a piece of work thus requires requesters to possess intimate crowdsourcing expertise (e.g., to control quality) and to "program" the crowd on their own. To make crowdsourcing more accessible, domain-specific service providers that know the domain requirements, tasks, and concerns and that can effectively assist also less skilled requesters in all phases of the crowdsourcing process are needed. CrowdFlower, for example, already positions itself as a Data Science platform; platforms for creative tasks like 99designs or Topcoder are domain-specific by design. Research on quality control has produced significant contributions so far, but it too needs to focus on domain-specific aspects if it wants to excel. Guaranteeing the quality of labels for images is just so different from doing so, for example, for text translations.
— *Crowd work regulation and ethics.* Crowdsourcing is a worldwide phenomenon and practice that allows workers and requesters from all over the world to establish business relationships. Yet, common rules and regulations for crowd work, for example, regarding taxation, pension, superannuation, resolution of disputes and similar, are still missing. Each crowdsourcing provider provides its own rules and legal protections so far, if at all. Similarly, the ecosystem as a whole (including workers, requesters, and platform providers alike) needs to grow shared work ethics and standards. MobileWorks, for instance, guarantees its workers

hourly wages, but this can only represent a starting point for sustainable crowd work. On the IT research side, controlling the compliance of rules, regulations, and ethical aspects may ask for novel monitoring and assessment techniques, for example, in the context of collusion detection.

REFERENCES

Ittai Abraham, Omar Alonso, Vasilis Kandylas, Rajesh Patel, Steven Shelford, and Aleksandrs Slivkins. 2016. How many workers to ask?: Adaptive exploration for collecting high quality labels. In *ACM SIGIR 2016*. 473–482. http://doi.acm.org/10.1145/2911451.2911514

Bo Thomas Adler and Luca De Alfaro. 2007. A content-driven reputation system for the Wikipedia. In *WWW 2007*. 261–270.

Bo Thomas Adler, Luca De Alfaro, Santiago M. Mola-Velasco, Paolo Rosso, and Andrew G. West. 2011. Wikipedia vandalism detection: Combining natural language, metadata, and reputation features. In *Computational Linguistics and Intelligent Text Processing*. Springer, 277–288.

Charu C. Aggarwal. 2013. An introduction to outlier analysis. In *Outlier Analysis*. Springer, 1–40.

Salman Ahmad, Alexis Battle, Zahan Malkani, and Sepander Kamvar. 2011. The Jabberwocky programming environment for structured social computing. In *UIST'11*. 53–64.

Luis von Ahn. 2006. Games with a purpose. *Computer* 39, 6 (June 2006), 92–94.

Harini Alagarai Sampath, Rajeev Rajeshuni, and Bipin Indurkhya. 2014. Cognitively inspired task design to improve user performance on crowdsourcing platforms. In *CHI 2014*. 3665–3674.

Mohammad Allahbakhsh, Boualem Benatallah, Aleksandar Ignjatovic, Hamid Reza Motahari-Nezhad, Elisa Bertino, and Schahram Dustdar. 2013. Quality control in crowdsourcing systems: Issues and directions. *IEEE Internet Computing* 17, 2 (March 2013), 76–81.

Mohammad Allahbakhsh, Aleksandar Ignjatovic, Boualem Benatallah, Seyed-Mehdi-Reza Beheshti, Elisa Bertino, and Norman Foo. 2012. Reputation management in crowdsourcing systems. In *CollaborateCom 2012*. 664–671.

Mohammad Allahbakhsh, Samira Samimi, Hamid Reza Motahari-Nezhad, and Boualem Benatallah. 2014. Harnessing implicit teamwork knowledge to improve quality in crowdsourcing processes. In *SOCA 2014*. 17–24.

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. 2012. Collaborative workflow for crowdsourcing translation. In *CSCW 2012*. 1191–1194.

Iheb Ben Amor, Salima Benbernou, Mourad Ouziri, Zaki Malik, and Brahim Medjahed. 2016. Discovering best teams for data leak-aware crowdsourcing in social networks. *ACM Transactions on the Web* 10, 1 (2016), Article 2, 27 pages.

Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2013. Steering user behavior with badges. In *WWW 2013*. 95–106.

Jesse Anderton, Maryam Bashir, Virgil Pavlu, and Javed A. Aslam. 2013. An analysis of crowd workers mistakes for specific and complex relevance assessment task. In *CIKM 2013*. ACM, 1873–1876.

Paul André, Robert E. Kraut, and Aniket Kittur. 2014. Effects of simultaneous and sequential work structures on distributed collaborative interdependent tasks. In *CHI 2014*. 139–148.

Donovan Artz and Yolanda Gil. 2007. A survey of trust in computer science and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web* 5, 2 (2007), 58–71.

Bahadir Ismail Aydin, Yavuz Selim Yilmaz, Yaliang Li, Qi Li, Jing Gao, and Murat Demirbas. 2014. Crowdsourcing for multiple-choice question answering. In *26th IAAI Conference*.

Piyush Bansal, Carsten Eickhoff, and Thomas Hofmann. 2016. Active content-based crowdsourcing task selection. In *CIKM 2016*. 529–538.

Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. 2009. Methodologies for data quality assessment and improvement. *ACM Computing Surveys (CSUR)* 41, 3 (2009), 16.

Michael S. Bernstein, David R. Karger, Robert C. Miller, and Joel Brandt. 2012. Analytic methods for optimizing realtime crowdsourcing. *CoRR* abs/1204.2995 (2012). http://arxiv.org/abs/1204.2995

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 *UIST 2010*. ACM, 313–322.

Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. 2010. VizWiz: Nearly real-time answers to visual questions. In *UIST 2010 (UIST'10)*. 333–342.

David Boud. 2013. *Enhancing Learning Through Self-assessment*. Routledge.

Ioannis Boutsis and Vana Kalogeraki. 2016. Location privacy for crowdsourcing applications. In *UbiComp 2016*. 694–705. DOI: http://dx.doi.org/10.1145/2971648.2971741

Alessandro Bozzon, Marco Brambilla, and Stefano Ceri. 2012. Answering search queries with crowdsearcher. In *WWW 2012*. 1009–1018.

Alessandro Bozzon, Marco Brambilla, Stefano Ceri, and Andrea Mauri. 2013. Reactive crowdsourcing. In *WWW 2013*. 153–164.

Jonathan Bragg, Daniel S. Weld, and others. 2013. Crowdsourcing multi-label classification for taxonomy creation. In *1st AAAI Conference on Human Computation and Crowdsourcing*.

Caleb Chen Cao, Lei Chen, and Hosagrahar Visvesvaraya Jagadish. 2014. From labor to trader: Opinion elicitation via online crowds as a market. In *KDD 2014*. 1067–1076.

Cinzia Cappiello, Florian Daniel, Agnes Koschmider, Maristella Matera, and Matteo Picozzi. 2011. A quality model for mashups. In *ICWE 2011*. 137–151. DOI: http://dx.doi.org/10.1007/978-3-642-22233-7_10

Ioannis Caragiannis, Ariel D. Procaccia, and Nisarg Shah. 2014. Modal ranking: A uniquely robust voting rule. In *AAAI 2014*. 616–622.

Ruggiero Cavallo and Shaili Jain. 2012. Efficient crowdsourcing contests. In *Proceedings of AAMAS 2012 - Volume 2*. 677–686.

Jesse Chandler, Gabriele Paolacci, and Pam Mueller. 2013. Risks and rewards of crowdsourcing marketplaces. In *Handbook of Human Computation*. Springer, 377–392.

Xi Chen, Qihang Lin, and Dengyong Zhou. 2013. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *ICML 2013*, Vol. 28, 64–72.

Justin Cheng, Jaime Teevan, and Michael S. Bernstein. 2015a. Measuring crowdsourcing effort with error-time curves. In *CHI 2015*. ACM, New York, 1365–1374.

Justin Cheng, Jaime Teevan, Shamsi T. Iqbal, and Michael S. Bernstein. 2015b. Break it down: A comparison of macro- and microtasks. In *CHI 2015*. 4061–4064. http://doi.acm.org/10.1145/2702123.2702146

Lydia B. Chilton, John J. Horton, Robert C. Miller, and Shiri Azenkot. 2010. Task search in a human computation market. In *HCOMP 2010*. 1–9. http://doi.acm.org/10.1145/1837885.1837889

Peng Dai, Jeffrey M. Rzeszotarski, Praveen Paritosh, and Ed H. Chi. 2015. And now for something completely different: Improving crowdsourcing workflows with micro-diversions. In *CSCW 2015*. ACM, New York, 628–638. DOI: http://dx.doi.org/10.1145/2675133.2675260

Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. 2013. Aggregating crowdsourced binary ratings. In *WWW 2013*. 285–294.

Martin Davtyan, Carsten Eickhoff, and Thomas Hofmann. 2015. Exploiting document content for efficient aggregation of crowdsourcing votes. In *CIKM 2015*. 783–790.

Luca De Alfaro, Ashutosh Kulshreshtha, Ian Pye, and Bo Thomas Adler. 2011. Reputation systems for open collaboration. *Communications of the ACM* 54, 8 (2011), 81–87.

Luca De Alfaro, Vassilis Polychronopoulos, and Michael Shavlovsky. 2015. Reliable aggregation of boolean crowdsourced tasks. In *HCOMP 2015*.

Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2013. Large-scale linked data integration using probabilistic reasoning and crowdsourcing. *The VLDB Journal* 22, 5 (2013), 665–687.

Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, and Philippe Cudré-Mauroux. 2014. Scaling-up the crowd: Micro-task pricing schemes for worker retention and latency improvement. In *HCOMP 2014*.

Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2012. Mechanical cheat: Spamming schemes and adversarial techniques on crowdsourcing platforms. In *CrowdSearch*. 26–30.

Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2013. Pick-a-crowd: Tell me what you like, and I'll tell you what to do. In *WWW 2013*. 367–374.

Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2016. Scheduling human intelligence tasks in multi-tenant crowd-powered systems. In *WWW 2016*. 855–865.

Mira Dontcheva, Robert R. Morris, Joel R. Brandt, and Elizabeth M. Gerber. 2014. Combining crowdsourcing and learning to improve engagement and performance. In *CHI 2014*. 3379–3388.

Christoph Dorn, R. N. Taylor, and S. Dustdar. 2012. Flexible Social Workflows: Collaborations as human architecture. *IEEE Internet Computing* 16, 2 (March 2012), 72–77.

Shayan Doroudi, Ece Kamar, Emma Brunskill, and Eric Horvitz. 2016. Toward a learning science for complex crowdsourcing tasks. In *CHI 2016*. ACM, New York, NY, USA, 2623–2634.

Steven Dow, Anand Kulkarni, Scott Klemmer, and Björn Hartmann. 2012. Shepherding the crowd yields better work. In *CSCW 2012*. 1013–1022.

Ryan Drapeau, Lydia B. Chilton, Jonathan Bragg, and Daniel S. Weld. 2016. MicroTalk: Using argumentation to improve crowdsourcing accuracy. In *HCOMP 2016*.

Carsten Eickhoff and Arjen P. de Vries. 2013. Increasing cheat robustness of crowdsourcing tasks. *Information Retrieval* 16, 2 (2013), 121–137.

Carsten Eickhoff, Christopher G. Harris, Arjen P. de Vries, and Padmini Srinivasan. 2012. Quality through flow and immersion: Gamifying crowdsourced relevance assessments. In *SIGIR 2012*. 871–880.

Kinda El Maarry, Ulrich Güntzer, and Wolf-Tilo Balke. 2015. A majority of wrongs doesn't make it right - On crowdsourcing quality for skewed domain tasks. In *WISE 2015*. 293–308.

Boi Faltings, Radu Jurca, Pearl Pu, and Bao Duy Tran. 2014. Incentives to counter bias in human computation. In *HCOMP 2014*. http://www.aaai.org/ocs/index.php/HCOMP/HCOMP14/paper/view/8945.

Meng Fang, Jie Yin, and Dacheng Tao. 2014. Active learning for crowdsourcing using knowledge transfer. In *28th AAAI Conference on Artificial Intelligence*.

Siamak Faradani, Björn Hartmann, and Panagiotis G. Ipeirotis. 2011. What's the right price? Pricing tasks for finishing on time.*Human Computation* 11 (2011).

Oluwaseyi Feyisetan and Elena Simperl. 2016. Please stay vs let's play: Social pressure incentives in paid collaborative crowdsourcing. In *ICWE 2016*. 405–412.

Oluwaseyi Feyisetan, Elena Simperl, Max Van Kleek, and Nigel Shadbolt. 2015. Improving paid microtasks through gamification and adaptive furtherance incentives. In *WWW 2015*. 333–343.

Linton C. Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

Ujwal Gadiraju, Ricardo Kawase, Stefan Dietze, and Gianluca Demartini. 2015. Understanding malicious behavior in crowdsourcing platforms: The case of online surveys. In *CHI 2015*, Vol. 15.

Snehalkumar (Neil) S. Gaikwad, Durim Morina, Adam Ginzberg, Catherine Mullings, Shirish Goyal, Dilrukshi Gamage, Christopher Diemert, Mathias Burton, Sharon Zhou, Mark Whiting, Karolina Ziulkoski, Alipta Ballav, Aaron Gilbee, Senadhipathige S. Niranga, Vibhor Sehgal, Jasmine Lin, Leonardy Kristianto, Angela Richmond-Fuller, Jeff Regino, Nalin Chhibber, Dinesh Majeti, Sachin Sharma, Kamila Mananova, Dinesh Dhakal, William Dai, Victoria Purynova, Samarth Sandeep, Varshine Chandrakanthan, Tejas Sarma, Sekandar Matin, Ahmed Nasser, Rohit Nistala, Alexander Stolzoff, Kristy Milland, Vinayak Mathur, Rajan Vaish, and Michael S. Bernstein. 2016. Boomerang: Rebounding the consequences of reputation feedback on crowdsourcing platforms. In *UIST 2016*. 625–637.

Chao Gao, Yu Lu, and Denny Zhou. 2016. Exact exponent in optimal rates for crowdsourcing. In *ICML 2016*. 603–611.

Mihai Georgescu, Dang Duc Pham, Claudiu S. Firan, Wolfgang Nejdl, and Julien Gaugaz. 2012. Map to humans and reduce error: Crowdsourcing for deduplication applied to digital libraries. In *CIKM 2012*. ACM, 1970–1974.

Derek L. Hansen, Patrick J. Schone, Douglas Corey, Matthew Reid, and Jake Gehring. 2013. Quality control mechanisms for crowdsourcing: Peer review, arbitration, & expertise at familysearch indexing. In *CSCW 2013*. 649–660.

Kotaro Hara, Vicki Le, and Jon Froehlich. 2013. Combining crowdsourcing and google street view to identify street-level accessibility problems. In *CHI 2013*. 631–640.

Jan Hartmann, Alistair Sutcliffe, and Antonella De Angeli. 2008. Towards a theory of user judgment of aesthetics and user interface quality. *ACM Transactions on Computer-Human Interaction* 15, 4 (2008), 15.

Kenji Hata, Ranjay Krishna, Li Fei-Fei, and Michael S. Bernstein. 2017. A glimpse far into the future: Understanding long-term crowd worker quality. In *CSCW 2017*. 889–901.

Jeffrey Heer and Michael Bostock. 2010. Crowdsourcing graphical perception: Using Mechanical Turk to assess visualization design. In *CHI 2010*. 203–212.

Kurtis Heimerl, Brian Gawalt, Kuang Chen, Tapan Parikh, and Björn Hartmann. 2012. CommunitySourcing: Engaging local crowds to perform expert work via physical kiosks. In *CHI 2012*. 1539–1548.

James Herbsleb, David Zubrow, Dennis Goldenson, Will Hayes, and Mark Paulk. 1997. Software quality and the capability maturity model. *Communications of the ACM* 40, 6 (1997), 30–40.

Paul Heymann and Hector Garcia-Molina. 2011. Turkalytics: Analytics for human computation. In *WWW 2011*. 477–486.

Chien-Ju Ho, Rafael Frongillo, and Yiling Chen. 2016. Eliciting categorical data for optimal aggregation. In *NIPS 2016*. Curran Associates, Inc., 2450–2458.

Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. 2015. Incentivizing high quality crowdwork. In *WWW 2015*. 419–429. DOI: http://dx.doi.org/10.1145/2736277.2741102

Chien-Ju Ho and Jennifer Wortman Vaughan. 2012. Online task assignment in crowdsourcing markets. In *AAAI*, Vol. 12. 45–51.

Simo Hosio, Jorge Goncalves, Vili Lehdonvirta, Denzil Ferreira, and Vassilis Kostakos. 2014. Situated crowdsourcing using a market model. In *UIST 2014*. ACM, 55–64.

Tobias Hossfeld, Christian Keimel, and Christian Timmerer. 2014. Crowdsourcing quality-of-experience assessments. *Computer* 47, 9 (Sept. 2014), 98–102.

Jeff. Howe. 2006. The rise of crowdsourcing. *Wired* (June 2006).

Chang Hu, Philip Resnik, Yakov Kronrod, and Benjamin Bederson. 2012. Deploying MonoTrans widgets in the wild. In *CHI 2012*. 2935–2938.

Shih-Wen Huang and Wai-Tat Fu. 2013a. Don't hide in the crowd!: Increasing social transparency between peer workers improves crowdsourcing outcomes. In *CHI 2013*. 621–630.

Shih-Wen Huang and Wai-Tat Fu. 2013b. Enhancing reliability using peer consistency evaluation in human computation. In *CSCW 2013*. 639–648.

Nguyen Quoc Viet Hung, Nguyen Thanh Tam, Ngoc Tran Lam, and Karl Aberer. 2013a. BATC: A benchmark for aggregation techniques in crowdsourcing. In *SIGIR 2013*. 1079–1080.

Nguyen Quoc Viet Hung, Nguyen Thanh Tam, Lam Ngoc Tran, and Karl Aberer. 2013b. An evaluation of aggregation techniques in crowdsourcing. In *WISE 2013*. Springer, 1–15.

Nguyen Quoc Viet Hung, Duong Chi Thang, Matthias Weidlich, and Karl Aberer. 2015. Minimizing efforts in validating crowd answers. In *SIGMOD 2015*. 999–1014.

Jane Hunter, Abdulmonem Alabri, and Catharina van Ingen. 2013. Assessing the quality and trustworthiness of citizen science data. *Concurrency and Computation: Practice and Experience* 25, 4 (2013).

Trung Dong Huynh, Mark Ebden, Matteo Venanzi, Sarvapali D. Ramchurn, Stephen J. Roberts, and Luc Moreau. 2013. Interpretation of crowdsourced activities using provenance network analysis. In *HCOMP 2013*.

Aleksandar Ignjatovic, Norman Foo, and Chung Tong Lee. 2008. An analytic approach to reputation ranking of participants in online transactions. In *WI/IAT 2008*. 587–590.

Kazushi Ikeda and Michael S. Bernstein. 2016. Pay it backward: Per-task payments on crowdsourcing platforms reduce productivity. In *CHI 2016*. 4111–4121. http://doi.acm.org/10.1145/2858036.2858327

Panagiotis G. Ipeirotis. 2010. Analyzing the Amazon Mechanical Turk marketplace. *XRDS* 17, 2 (Dec. 2010), 16–21.

Panagiotis G. Ipeirotis and Evgeniy Gabrilovich. 2014. Quizz: Targeted crowdsourcing with a billion (potential) users. In *WWW 2014*. 143–154.

Lilly C. Irani and M. Silberman. 2013. Turkopticon: Interrupting worker invisibility in Amazon Mechanical Turk. In *CHI 2013*. 611–620.

Srikanth Jagabathula, Lakshminarayanan Subramanian, and Ashwin Venkataraman. 2014. Reputation-based worker filtering in crowdsourcing. In *NIPS 2014*. Curran Associates, Inc., 2492–2500.

Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. 2013. Evaluating the crowd with confidence. In *KDD 2013*. ACM, 686–694.

Oliver P. John, Laura P. Naumann, and Christopher J. Soto. 2008. Paradigm shift to the integrative big five trait taxonomy. *Handbook of Personality: Theory and Research* (3rd ed.). Guilford Press, New York, 114–158.

Oliver P. John and Sanjay Srivastava. 1999. The big five trait taxonomy: History, measurement, and theoretical perspectives. *Handbook of Personality: Theory and Research* (2nd ed.). Guilford Press, New York, 102–138.

Hyun Joon Jung and Matthew Lease. 2011. Improving consensus accuracy via Z-score and weighted voting. In *Human Computation*.

Hyun Joon Jung and Matthew Lease. 2012. Inferring missing relevance judgments from crowd workers via probabilistic matrix factorization. In *SIGIR 2012*. 1095–1096.

Hyun Joon Jung, Yubin Park, and Matthew Lease. 2014. Predicting next label quality: A time-series model of crowdwork. In *HCOMP 2014*.

Ho-Won Jung, Seung-Gweon Kim, and Chang-Shin Chung. 2004. Measuring software product quality: A survey of ISO/IEC 9126. *IEEE Software* 5 (2004), 88–92.

Sanjay Kairam and Jeffrey Heer. 2016. Parting crowds: Characterizing divergent interpretations in crowdsourced annotation tasks. In *CSCW 2016*. 1637–1648. DOI: http://dx.doi.org/10.1145/2818048.2820016

David R. Karger, Sewoong Oh, and Devavrat Shah. 2011. Iterative learning for reliable crowdsourcing systems. In *NIPS 2011*. Curran Associates, Inc., 1953–1961.

David R. Karger, Sewoong Oh, and Devavrat Shah. 2014. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research* 62, 1 (2014), 1–24.

Geoff Kaufman, Mary Flanagan, and Sukdith Punjasthitkul. 2016. Investigating the impact of 'emphasis frames' and social loafing on player motivation and performance in a crowdsourcing game. In *CHI 2016*. 4122–4128.

Gabriella Kazai, Jaap Kamps, Marijn Koolen, and Natasa Milic-Frayling. 2011. Crowdsourcing for book search evaluation: Impact of hit design on comparative system ranking. In *SIGIR 2011*. 205–214.

Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. 2011. Worker types and personality traits in crowdsourcing relevance labels. In *CIKM 2011*. ACM, 1941–1944.

Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. 2012. The face of quality in crowdsourcing relevance labels: Demographics, personality and labeling accuracy. In *CIKM 2012*. ACM, 2583–2586.

Gabriella Kazai and Imed Zitouni. 2016. Quality management in crowdsourcing using gold judges behavior. In *WSDM 2016*. 267–276. DOI: http://dx.doi.org/10.1145/2835776.2835835

Robert Kern, Hans Thies, Cordula Bauer, and Gerhard Satzger. 2010. Quality assurance for human-based electronic services: A decision matrix for choosing the right approach. In *ICWE 2010 Workshops*. 421–424.

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 *1st ACM Symposium on Computing for Development*. ACM, 12.

Roman Khazankin, Daniel Schall, and Schahram Dustdar. 2012. Predicting QoS in scheduled crowdsourcing. In *CAISE 2012*. 460–472.

Ashiqur R. KhudaBukhsh, Jaime G. Carbonell, and Peter J. Jansen. 2014. Detecting non-adversarial collusion in crowdsourcing. In *HCOMP 2014*.

Aniket Kittur. 2010. Crowdsourcing, collaboration and creativity. *ACM Crossroads* 17, 2 (2010), 22–26.

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*. ACM, 453–456.

Aniket Kittur, Susheel Khamkar, Paul André, and Robert Kraut. 2012. CrowdWeaver: Visually managing complex crowd work. In *CSCW 2012*. 1033–1036.

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 *CSCW 2013*. 1301–1318.

Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. 2011. Crowdforge: Crowdsourcing complex work. In *UIST'11*. 43–52.

Masatomo Kobayashi, Shoma Arita, Toshinari Itoko, Shin Saito, and Hironobu Takagi. 2015. Motivating multi-generational crowd workers in social-purpose work. In *CSCW 2015*. 1813–1824.

Ari Kobren, Chun How Tan, Panagiotis Ipeirotis, and Evgeniy Gabrilovich. 2015. Getting more for less: Optimized crowdsourcing with dynamic tasks and goals. In *WWW 2015*. 592–602.

Markus Krause and René F. Kizilcec. 2015. To play or not to play: Interactions between response quality and task complexity in games and paid crowdsourcing. In *HCOMP 2015*. 102–109.

Ranjay A. Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A. Shamma, Li Fei-Fei, and Michael S. Bernstein. 2016. Embracing error to enable rapid crowdsourcing. In *CHI 2016*. 3167–3179.

Kyriakos Kritikos, Barbara Pernici, Pierluigi Plebani, Cinzia Cappiello, Marco Comuzzi, Salima Benrernou, Ivona Brandic, Attila Kertész, Michael Parkin, and Manuel Carro. 2013. A survey on service quality description. *ACM Computing Surveys (CSUR)* 46, 1 (2013), 1.

Pavel Kucherbaev, Florian Daniel, Stefano Tranquillini, and Maurizio Marchese. 2016b. Crowdsourcing processes: A survey of approaches and opportunities. *IEEE Internet Computing* 20, 2 (2016), 50–56.

Pavel Kucherbaev, Florian Daniel, Stefano Tranquillini, and Maurizio Marchese. 2016a. ReLauncher: Crowdsourcing microtasks runtime controller. In *CSCW 2016*. 1607–1612.

Anand Kulkarni, Matthew Can, and Björn Hartmann. 2012a. Collaboratively crowdsourcing workflows with Turkomatic. In *CSCW'12*. ACM, New York, 1003–1012.

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

Anand Kulkarni, Prayag Narula, David Rolnitzky, and Nathan Kontny. 2014. Wish: Amplifying creative ability with expert crowds. In *HCOMP 2014*.

Walter S. Lasecki, Christopher D. Miller, and Jeffrey P. Bigham. 2013. Warping time for more effective real-time crowdsourcing. In *CHI 2013*. 2033–2036.

Walter S. Lasecki, Jeffrey M. Rzeszotarski, Adam Marcus, and Jeffrey P. Bigham. 2015. The effects of sequence and delay on crowd work. In *CHI 2015*. 1375–1378.

Walter S. Lasecki, Young Chol Song, Henry Kautz, and Jeffrey P. Bigham. 2013. Real-time crowd labeling for deployable activity recognition. In *CSCW 2013*. 1203–1212.

Walter S. Lasecki, Jaime Teevan, and Ece Kamar. 2014. Information extraction and manipulation threats in crowd-powered systems. In *CSCW 2014*. ACM, 248–256.

Paolo Laureti, Lionel Moret, Yi-Cheng Zhang, and Yi-Kuo Yu. 2006. Information filtering via iterative refinement. *Europhysics Letters* 75 (2006), 1006.

Edith Law, Ming Yin, Joslin Goh, Kevin Chen, Michael A. Terry, and Krzysztof Z. Gajos. 2016. Curiosity killed the cat, but makes crowdwork better. In *CHI 2016*. ACM, New York, 4098–4110.

John Le, Andy Edmonds, Vaughn Hester, and Lukas Biewald. 2010. Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution. In *SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation*. 21–26.

Robert C. Lewis and Bernhard H. Booms. 1983. *Emerging Perspectives on Service Marketing*. American Marketing, 99–107.

Hongwei Li, Bo Zhao, and Ariel Fuxman. 2014. The wisdom of minority: Discovering and targeting the right group of workers for crowdsourcing. In *WWW 2014*. 165–176.

Qi Li, Fenglong Ma, Jing Gao, Lu Su, and Christopher J. Quinn. 2016. Crowdsourcing high quality labels with a tight budget. In *WSDM 2016*. 237–246.

Christopher H. Lin, Ece Kamar, and Eric Horvitz. 2014. Signals in the silence: Models of implicit feedback in a recommendation system for crowdsourcing. In *AAAI 2014*. 908–915.

Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2010b. Exploring iterative and parallel human computation processes. In *ACM SIGKDD Workshop on Human Computation*. 68–76.

Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2010a. Turkit: Human computation algorithms on Mechanical Turk. In *UIST 2010*. ACM, 57–66.

Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2010c. Turkit: Human computation algorithms on Mechanical Turk. In *UIST'10*. ACM, New York, 57–66.

Chao Liu and Yi-Min Wang. 2012. TrueLabel + confusions: A spectrum of probabilistic models in analyzing multiple ratings.. In *ICML 2012*. icml.cc/ Omnipress.

Qiang Liu, Alexander T. Ihler, and Mark Steyvers. 2013. Scoring workers in crowdsourcing: How many control questions are enough? In *NIPS 2013*. Curran Associates, Inc., 1914–1922.

Benjamin Livshits and Todd Mytkowicz. 2014. Saving money while polling with interpoll using power analysis. In *HCOMP 2014*.

Thomas W. Malone, Robert Laubacher, and Chrysanthos Dellarocas. 2010. The collective intelligence genome. *IEEE Engineering Management Review* 38, 3 (2010), 38.

Andrew Mao, Ece Kamar, Yiling Chen, Eric Horvitz, Megan E. Schwamb, Chris J. Lintott, and Arfon M. Smith. 2013. Volunteering versus work for pay: Incentives and tradeoffs in crowdsourcing. In *HCOMP 2013*.

Adam Marcus, David Karger, Samuel Madden, Robert Miller, and Sewoong Oh. 2012. Counting with the crowd. In *Proceedings of the VLDB Endowment*, Vol. 6. VLDB Endowment, 109–120.

Elaine Massung, David Coyle, Kirsten F. Cater, Marc Jay, and Chris Preist. 2013. Using crowdsourcing to support pro-environmental Community Activism. In *CHI 2013*. 371–380.

Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. 2016. Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing. In *WWW 2016*. 843–853.

Tyler McDonnell, Matthew Lease, Mucahid Kutlu, and Tamer Elsayed. 2016. Why is that relevant? Collecting annotator rationales for relevance judgments. In *HCOMP 2016*.

Patrick Minder and Abraham Bernstein. 2012. Crowdlang: A programming language for the systematic exploration of human computation systems. In *Social Informatics*. Springer, 124–137.

Aliaksei Miniukovich and Antonella De Angeli. 2015. Visual diversity and user interface quality. In *British HCI 2015*. 101–109.

Kaixiang Mo, Erheng Zhong, and Qiang Yang. 2013. Cross-task crowdsourcing. In *KDD 2013*. 677–685.

Robert R. Morris, Mira Dontcheva, and Elizabeth M. Gerber. 2012. Priming for better performance in microtask crowdsourcing environments. *IEEE Internet Computing* 16, 5 (Sept. 2012), 13–19.

Yashar Moshfeghi, Alvaro F. Huertas-Rosero, and Joemon M. Jose. 2016. Identifying careless workers in crowdsourcing platforms: A game theory approach. In *ACM SIGIR 2016*. 857–860.

Swaprava Nath, Pankaj Dayama, Dinesh Garg, Y. Narahari, and James Y. Zou. 2012. Threats and trade-offs in resource critical crowdsourcing tasks over networks. In *AAAI 2012*.

Edward Newell and Derek Ruths. 2016. How one microtask affects another. In *CHI 2016*. 3155–3166.

Dong Nguyen, Dolf Trieschnigg, and Mariët Theune. 2014. Using crowdsourcing to investigate perception of narrative similarity. In *CIKM 2014*. ACM, 321–330.

Jakob Nielsen, Marie Tahir, and Marie Tahir. 2002. *Homepage Usability: 50 Websites Deconstructed*. Vol. 50. New Riders Indianapolis, IN.

Evangelos Niforatos, Ivan Elhart, and Marc Langheinrich. 2016. WeatherUSI: User-based weather crowdsourcing on public displays. In *ICWE 2016*. 567–570.

Jon Noronha, Eric Hysen, Haoqi Zhang, and Krzysztof Z. Gajos. 2011. Platemate: Crowdsourcing nutritional analysis from food photographs. In *UIST 2011*. ACM, 1–12.

Besmira Nushi, Adish Singla, Anja Gruenheid, Erfan Zamanian, Andreas Krause, and Donald Kossmann. 2015. Crowd access path optimization: Diversity matters. In *HCOMP 2015*.

Jungseul Ok, Sewoong Oh, Jinwoo Shin, and Yung Yi. 2016. Optimality of belief propagation for crowdsourced classification. In *ICML 2016*. JMLR.org, 535–544.

David Oleson, Alexander Sorokin, Greg P. Laughlin, Vaughn Hester, John Le, and Lukas Biewald. 2011. Programmatic gold: Targeted and scalable quality assurance in crowdsourcing. *HCOMP 2011* 11, 11 (2011).

Jasper Oosterman and Geert-Jan Houben. 2016. On the invitation of expert contributors from online communities for knowledge crowdsourcing tasks. In *ICWE 2016*. 413–421.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: Bringing order to the Web. (1999).

Chris Preist, Elaine Massung, and David Coyle. 2014. Competing or aiming to be average?: Normification as a means of engaging digital volunteers. In *CSCW 2014*. 1222–1233.

Cindy Puah, Ahmad Zaki Abu Bakar, and Chu Wei Ching. 2011. Strategies for community based crowdsourcing. In *ICRIIS 2011*. 1–4.

Alexander J. Quinn and Benjamin B. Bederson. 2014. AskSheet: Efficient human computation for decision making with spreadsheets. In *CSCW 2014*. 1456–1466.

Goran Radanovic and Boi Faltings. 2016. Learning to scale payments in crowdsourcing with properboost. In *HCOMP 2016*.

Karthikeyan Rajasekharan, Aditya P. Mathur, and See-Kiong Ng. 2013. Effective crowdsourcing for software feature ideation in online co-creation forums. In *SEKE 2013*. 119–124.

Huaming Rao, Shih-Wen Huang, and Wai-Tat Fu. 2013. What will others choose? How a majority vote reward scheme can improve human computation in a spatial location identification task. In *HCOMP 2013*.

Vikas C. Raykar and Shipeng Yu. 2011. Ranking annotators for crowdsourced labeling tasks. In *NIPS 2011*. Curran Associates Inc., 1809–1817.

Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S. Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S. Bernstein. 2014. Expert crowdsourcing with flash teams. In *UIST*. ACM, 75–85.

Jakob Rogstadius, Vassilis Kostakos, Aniket Kittur, Boris Smus, Jim Laredo, and Maja Vukovic. 2011. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. In *ICWSM*.

Markus Rokicki, Sergiu Chelaru, Sergej Zerr, and Stefan Siersdorfer. 2014. Competitive game designs for improving the cost effectiveness of crowdsourcing. In *CICM 2014*. ACM, 1469–1478.

Markus Rokicki, Sergej Zerr, and Stefan Siersdorfer. 2015. Groupsourcing: Team competition designs for crowdsourcing. In *WWW 2015*. 906–915.

Senjuti Basu Roy, Ioanna Lykourentzou, Saravanan Thirumuruganathan, Sihem Amer-Yahia, and Gautam Das. 2015. Task assignment optimization in knowledge-intensive crowdsourcing. *The VLDB Journal* 24, 4 (2015), 467–491.

Jeffrey M. Rzeszotarski and Aniket Kittur. 2011. Instrumenting the crowd: Using implicit behavioral measures to predict task performance. In *UIST 2011*. ACM, 13–22.

Jeffrey M. Rzeszotarski and Aniket Kittur. 2012. CrowdScape: Interactively visualizing user behavior and output. In *UIST 2012*. ACM, 55–62.

Yuko Sakurai, Tenda Okimoto, Masaaki Oka, Masato Shinoda, and Makoto Yokoo. 2013. Ability grouping of crowd workers via reward discrimination. In *HCOMP 2013*.

Benjamin Satzger, Harald Psaier, Daniel Schall, and Schahram Dustdar. 2013. Auction-based crowdsourcing supporting skill management. *Information Systems* 38, 4 (June 2013), 547–560.

Ognjen Scekic, Hong-Linh Truong, and Schahram Dustdar. 2013a. Incentives and rewarding in social computing. *Communications of the ACM* 56, 6 (2013), 72–82.

Ognjen Scekic, Hong-Linh Truong, and Schahram Dustdar. 2013b. Programming incentives in information systems. In *Advanced Information Systems Engineering*. Springer, 688–703.

Daniel Schall, Benjamin Satzger, and Harald Psaier. 2014. Crowdsourcing tasks to social networks in BPEL4People. *World Wide Web* 17, 1 (2014), 1–32.

Daniel Schall, Florian Skopik, and Schahram Dustdar. 2012. Expert discovery and interactions in mixed service-oriented systems. *IEEE Transactions on Services Computing* 5, 2 (2012), 233–245.

Thimo Schulze, Dennis Nordheimer, and Martin Schader. 2013. Worker perception of quality assurance mechanisms in crowdsourcing and human computation markets. In *AMCIS 2013*.

Nihar Bhadresh Shah and Denny Zhou. 2015. Double or nothing: Multiplicative incentive mechanisms for crowdsourcing. In *NIPS 2015*. Curran Associates, Inc., 1–9.

Nihar Bhadresh Shah and Dengyong Zhou. 2016. No oops, you won't do it again: Mechanisms for self-correction in crowdsourcing. In *ICML 2016*. 1–10.

Aashish Sheshadri and Matthew Lease. 2013. SQUARE: A benchmark for research on computing crowd consensus. In *HCOMP 2013*.

Yaron Singer and Manas Mittal. 2013. Pricing mechanisms for crowdsourcing markets. In *WWW*. 1157–1166.

Adish Singla, Ilija Bogunovic, Gábor Bartók, Amin Karbasi, and Andreas Krause. 2014. Near-optimally teaching the crowd to classify. In *ICML 2014*. JMLR.org, II-154–II-162.

Klaas-Jan Stol and Brian Fitzgerald. 2014. Two's company, three's a crowd: A case study of crowdsourcing software development. In *ICSE 2014*. 187–198.

Yu-An Sun and Christopher Dance. 2012. When majority voting fails: Comparing quality assurance methods for noisy human computation environment. *arXiv:1204.3516* (2012).

James Surowiecki. 2005. *The Wisdom of Crowds*. Anchor Books.

Oksana Tokarchuk, Roberta Cuel, and Marco Zamarian. 2012. Analyzing crowd labor and designing incentives for humans in the loop. *IEEE Internet Computing* 16, 5 (Sept. 2012), 45–51.

Lisa Torrey and Jude Shavlik. 2009. Transfer learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques* 1 (2009), 242.

Long Tran-Thanh, Trung Dong Huynh, Avi Rosenfeld, Sarvapali D. Ramchurn, and Nicholas R. Jennings. 2015. Crowdsourcing complex workflows under budget constraints. In *AAAI 2015*. 1298–1304.

Antti Ukkonen, Behrouz Derakhshan, and Hannes Heikinheimo. 2015. Crowdsourced nonparametric density estimation using relative distances. In *HCOMP 2015*.

Rajan Vaish, Keith Wyngarden, Jingshu Chen, Brandon Cheung, and Michael S. Bernstein. 2014. Twitch crowdsourcing: Crowd contributions in short bursts of time. In *CHI 2014*. 3645–3654.

Norases Vesdapunt, Kedar Bellare, and Nilesh Dalvi. 2014. Crowdsourcing algorithms for entity resolution. *Proceedings of the VLDB Endowment* 7, 12 (2014), 1071–1082.

Fernanda B. Viégas, Martin Wattenberg, and Matthew M. McKeon. 2007. The hidden order of Wikipedia. *Online Communities and Social Computing*. Springer, 445–454.

Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. 2008. reCAPTCHA: Human-based character recognition via web security measures. *Science* 321, 5895 (2008), 1465–1468.

Maja Vukovic and Claudio Bartolini. 2010. Towards a research agenda for enterprise crowdsourcing. *Leveraging Applications of Formal Methods, Verification, and Validation*. Springer, 425–434.

Maja Vukovic, Mariana Lopez, and Jim Laredo. 2010. Peoplecloud for the globally integrated enterprise. In *ICSOC/ServiceWave 2009 Workshops on Service-Oriented Computing*. Springer, 109–114.

Bo Waggoner and Yiling Chen. 2014. Output agreement mechanisms and common knowledge. In *HCOMP 2014*.

Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2012. Serf and turf: Crowdturfing for fun and profit. In *WWW 2012*. 679–688.

Fabian L. Wauthier and Michael I. Jordan. 2011. Bayesian bias mitigation for crowdsourcing. In *NIPS 2011*. Curran Associates, Inc., 1800–1808.

Mark E. Whiting, Dilrukshi Gamage, Snehalkumar (Neil) S. Gaikwad, Aaron Gilbee, Shirish Goyal, Alipta Ballav, Dinesh Majeti, Nalin Chhibber, Angela Richmond-Fuller, Freddie Vargus, Tejas Seshadri Sarma, Varshine Chandrakanthan, Teogenes Moura, Mohamed Hashim Salih, Gabriel Bayomi Tinoco Kalejaiye, Adam Ginzberg, Catherine A. Mullings, Yoni Dayan, Kristy Milland, Henrique Orefice, Jeff Regino, Sayna Parsi, Kunz Mainali, Vibhor Sehgal, Sekandar Matin, Akshansh Sinha, Rajan Vaish, and Michael S. Bernstein. 2017. Crowd guilds: Worker-led reputation and feedback on crowdsourcing platforms. In *CSCW 2017*. 1902–1913. DOI: http://dx.doi.org/10.1145/2998181.2998234

Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. 2012. Strategies for crowdsourcing social data analysis. In *CHI 2012*. 227–236.

Stephen M. Wolfson and Matthew Lease. 2011. Look before you leap: Legal pitfalls of crowdsourcing. *Proceedings of the American Society for Information Science and Technology* 48, 1 (2011), 1–10.

Yan Yan, Glenn M. Fung, Rómer Rosales, and Jennifer G. Dy. 2011. Active learning from crowds. In *ICML 2011*. 1161–1168.

Jie Yang, Judith Redi, Gianluca DeMartini, and Alessandro Bozzon. 2016. Modeling task complexity in crowdsourcing. In *HCOMP 2016*. 249–258.

Ming Yin, Yiling Chen, and Yu-An Sun. 2014. Monetary interventions in crowdsourcing task switching. In *HCOMP 2014*.

Lixiu Yu, Paul André, Aniket Kittur, and Robert Kraut. 2014. A comparison of social, learning, and financial strategies on crowd engagement and output quality. In *CSCW 2014*. 967–978.

Yi-Kuo Yu, Yi-Cheng Zhang, Paolo Laureti, and Lionel Moret. 2006. Decoding information from noisy, redundant, and intentionally distorted sources. *Physica A: Statistical Mechanics and its Applications* 371, 2 (2006), 732–744.

Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. 2015. TaskRec: A task recommendation framework in crowdsourcing systems. *Neural Processing Letters* 41, 2 (2015), 223–238.

Jing Zhang, Xindong Wu, and Victor S. Sheng. 2015. Imbalanced multiple noisy labeling. *IEEE Transactions on Knowledge & Data Engineering* 27, 2 (2015), 489–503.

Zhou Zhao, Da Yan, Wilfred Ng, and Shi Gao. 2013. A transfer learning based framework of crowd-selection on twitter. In *KDD 2013*. ACM, 1514–1517.

Haiyi Zhu, Steven P. Dow, Robert E. Kraut, and Aniket Kittur. 2014. Reviewing versus doing: Learning and performance in crowd assessment. In *CSCW 2014*. 1445–1455.

Honglei Zhuang and Joel Young. 2015. Leveraging in-batch annotation bias for crowdsourced active learning. In *WSDM 2015*. 243–252. DOI: http://dx.doi.org/10.1145/2684822.2685301

Received June 2016; revised September 2017; accepted September 2017