UNIVERSITÀ DEGLI STUDI
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE
ICT International Doctoral School

Quality Assurance Strategies in Microtask Crowdsourcing

Pavel Kucherbaev

Advisor

Prof. Maurizio Marchese

University of Trento

Co-Advisor

Prof. Florian Daniel

Politecnico di Milano

March 2016

Acknowledgments

Five years ago I was not really sure what a PhD is. I would like to thank Prof. Fabio Casati, Dr. Roman Chirikov and Prof. Nafisa Yusupova for giving me the first ideas about the PhD and encouraging to start it at the University of Trento.

In the very beginning my research activities went not very smooth and I thought about quitting my PhD every 4 months during the first year, partly because of the great course InnovAction Lab given by Augusto Coppola, whom I also thank for framing my entrepreneurial mindset. I thank Prof. Fabio Casati for not kicking me out of the doctoral school during this time and for helping me to find my path.

All the following academic work happened in a close collaboration and under the great mentorship of Prof. Maurizio Marchese, Prof. Florian Daniel and Dr. Stefano Tranquillini. I started to like doing research because of you. Thank you very much Florian for those simple life truths you used to share with me.

University of Trento, European Institute of Innovation & Technology and Trento Rise provided me with all the possible help and support. I would like to thank Italian and European tax-payers for supporting my PhD.

I would like to thank Lukas Biewald for providing me with an opportunity to have a great internship at CrowdFlower in San Francisco.

I would like to thank my friends and colleagues for fulfilling my life and giving me professional advices and critical comments. I would like to thank my family and my wife who always encourage me to start my new ventures giving me both support and freedom.

Pavel

Abstract

Crowdsourcing is the outsourcing of a unit of work to a crowd of people via an open call for contributions. While there are various forms of crowdsourcing, such as open innovation, civic engagement and crowdfunding in this work we specifically focus on microtasking. Microtasking is a branch of crowdsourcing, where a work is presented as a set of identical microtasks, each requiring contributors only several minutes to complete usually in exchange for a reward of less than 1 USD. Labeling images, transcribing documents, analyzing sentiments of short sentences and cleaning datasets are popular examples of work which could be solved as microtasks.

Available up to date microtask crowdsourcing platforms, such as CrowdFlower and Amazon Mechanical Turk, allow thousands of microtasks to be solved in parallel by hundreds of contributors available online. Nothing or little is known about these contributors. They have no legal obligations to perform tasks well and in time. One of the biggest problems in microtask crowdsourcing is assuring that only responses of high quality are collected in a short period of time, while responses of low quality and those given by workers not following instructions are eliminated.

To tackle the problem of quality in microtask crowdsourcing, it is necessary to study different quality attributes, to investigate what causes low quality of results and slow task execution in microtask crowdsourcing, to identify effective methods to both assess and assure that these quality attributes are of high level.

We conducted the most extensive literature review analysis of quality attributes, assessment and assurance techniques ever done in the area of microtasking and crowdsourcing in general. We further advanced the state of the art in three research tracks: i) Improving accuracy and execution speed (the major track), where we monitor in-page user activity of each individual worker, automatically predict abandoned assignments causing delays and assignments with low quality of results, and relaunch them to other workers using our tool ReLauncher; ii) Crowdsourcing complex processes, where we introduce BPMN-extensions to design business processes of both crowd and machine tasks, and the crowdsourcing platform Crowd Computer to deploy these tasks; and iii) Improving workers user experience, where we identify problems workers face searching for tasks to work on, address these problems in our prototype of the task listing interface and introduce a new mobile crowdsourcing platform, CrowdCafe, designed in a way to optimize task searching time and to motivate workers with tangible rewards, such as a coffee.

Keywords

Crowdsourcing, Microtasks, Quality Control, Accuracy, Execution Speed

Contents

Executive summary

This thesis is structured as an executive summary of the work carried out. In this executive summary we give an introduction to the field of microtask crowdsourcing, define problems of quality control, provide an extensive literature review of quality control in crowdsourcing, discuss three research tracks which we followed and summarize contributions we made. While the primary focus of this PhD is improving accuracy and overall execution speed in microtask crowdsourcing, we also conducted research about ways to crowdsource complex processes and ways to improve workers experience on crowdsourcing platforms. All these tracks are interconnected and contribute to the common topic of quality control in microtask crowdsourcing. We finish this executive summary with a discussion of limitations and also our vision about the future of microtask crowdsourcing platforms. This vision is based on more than 3 years of research, occasional experience of being both workers and requesters, and an experience gained at an internship at CrowdFlower – one of the major microtask crowdsourcing platforms available.

Most of the results we got from our work are already published in international conferences and journals, those which are not yet published are available as pre-prints. All these publications along with the technical report included in Appendix are cited and discussed in this thesis.

1 Crowdsourcing

James Surowiecki discussed in his book [114] an example where at a country fair people gave guesses for a weight of an ox. The mean of everybody's guesses was 1197 pounds, while the actual weight appeared to be 1198. This phenomenon, when an average of predictions given by a large crowd is better than a prediction of any individual in this crowd, is called "wisdom of crowds". The example of this phenomenon is not unique and there are even companies, such as Cultivatelabs[1], utilizing it for making market predictions. While many people refer to the terms "crowdsourcing" and "wisdom of crowds" interchangeably, it is accurate to refer to the latter one only as a special case of crowdsourcing.

Crowdsourcing was first coined in 2006 in the magazine Wired by a journalist Jeff Howe as an approach for outsourcing a unit of work to a crowd of people via an open call for contributions [49]. The approach existed long before Howe's article, still thanks to the Internet it became much easier to reach large groups of people and to outsource work to them. It is not yet clear whether this term brought more clarity or confusion into the academic and industrial communities. Estelles et al. did an extensive analysis of different definitions given in academic publications [35].

Crowdsourcing is a high level term. It has different forms depending on the type of work, the way this work is delivered to workers, the way workers perform this job and the way workers are evaluated and compensated. Usually each form has its own name. When people volunteer doing tasks provided by scientific communities at Zooniverse (`https://www.zooniverse.org/`) it is called *citizen science*, when people pledge money for projects they like or pre-order goods on Kickstarter (`https://www.kickstarter.com/`) it is called *crowdfunding*, when people sign po-

[1] `https://www.cultivatelabs.com`

litical petitions on Change.org (`https://www.change.org/`) it is called *civic engagement*, when people propose solutions for complex challenges on Innocentive (`http://innocentive.com/`) it is called *social innovation*. This list is not complete and a more detailed one can be found in the book "Getting Results from Crowds" [24]. In this thesis we primarily focus on *microtasking*.

1.1 Microtask crowdsourcing

Microtasking is another branch of crowdsourcing, where requesters publish their work on a crowdsourcing platform in a form of identical microtasks, each requiring online contributors – workers – only several minutes to complete, usually in exchange for a reward of less than 1 USD. Labeling images, transcribing documents, analyzing sentiment of short sentences, cleaning datasets are all popular examples of work which could be solved in a form of microtasks. While there is the perception that businesses use microtasking because it is cheap, they also use it because it provides scalability, diversity and availability 24/7 [24]. Some time ago Instagram used microtask crowdsourcing to check all images for adult content. Now during the time of big interest in machine learning many companies and individuals use microtask crowdsourcing platforms to generate human labels for their training datasets. This trend is supported by the fact that one of the biggest microtask labor providers, CrowdFlower, shifted its focus from being a general purpose crowdsourcing platform to a data science tool for cleaning datasets. Another popular use case of microtask crowdsourcing: is the so-called *lead generation* – collection of information about prospective customers. MobileWorks originally started as a general purpose crowdsourcing platform providing fair wages for people from developing countries and converted now into the domain specific platform Lead Genius focused on lead generation. Many small startups use microtask

crowdsourcing as "Artificial Artificial Intelligence", when users think that features are performed automatically, while there are real people behind their execution.

1.2 Platforms for microtasking

A detailed analysis of microtask crowdsourcing platforms was carried out by Vakharia et al. [119]. Here below we provide a brief overview of some of them: Amazon Mechanical Turk (MTurk), CrowdFlower, MicroWorkers and CloudFactory.

Amazon Mechanical Turk (`http://mturk.com/`) – was originally built by Amazon for internal use to classify goods in their catalog and was later publicly released in 2005. Since that year the platform is still in its beta version and it did not experience significant changes since its release. This platform is extensively used by the research community and most of the experiments discussed in academic publications were conducted on MTurk. According to the information provided on the platform website, there are around 200 000 microtasks available on MTurk at any given moment. Not much else is know about properties of this market. Thanks to the continuous study carried since 2009 by Difallah et al. [28] by periodically crawling the platform task list more is known about its dynamics. According to this study the most popular task type nowadays is audio transcription, the most popular reward amount is 0.05 USD per microtask and surveys are the most popular microtask restricted to US-based workers. In Figure 1.1 we show an example of the task listing page in Amazon Mechanical Turk used by workers to search for tasks to work on.

CrowdFlower (`https://crowdflower.com`) – was released in 2007 as a tool on top of MTurk to support some quality control features, such as

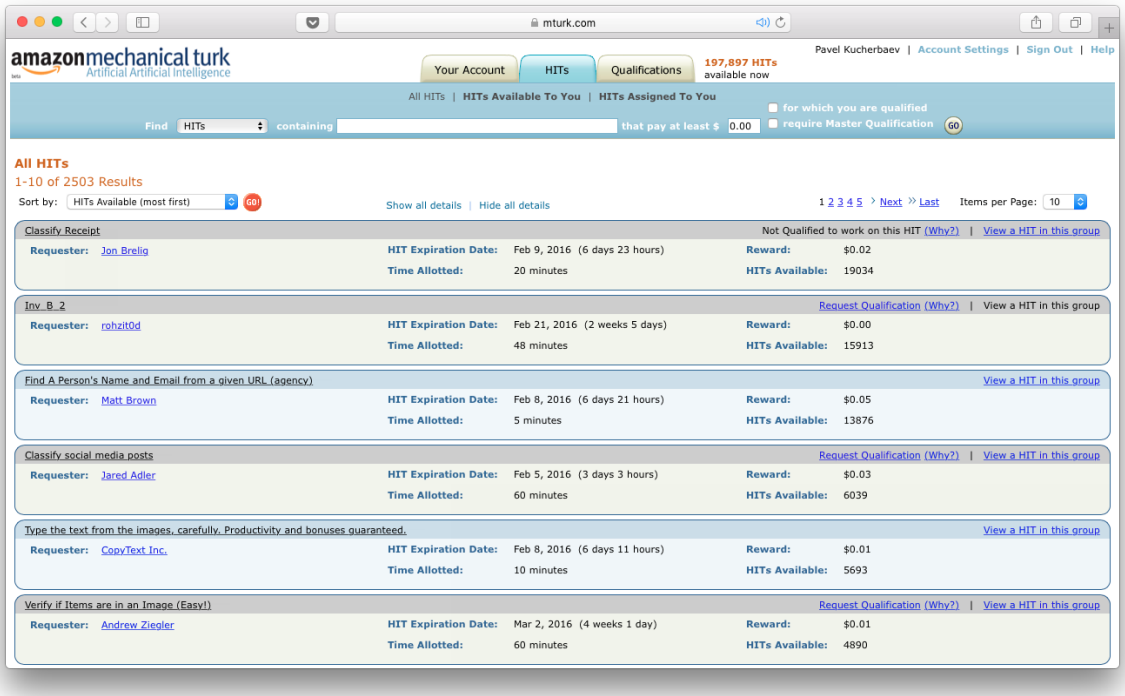

Figure 1.1: The task listing page in Amazon Mechanical Turk

results aggregation. Later CrowdFlower[2] made it possible for 3rd party platforms – channels – to integrate its task listing and execution interface, so CrowdFlower could broaden its worker base while the channels received a commission as reward. . In such a way CrowdFlower managed to acquire a user base of several million registered users performing tasks published on the platform (among which about 22000 are active per day according to the report [93]). In Figure 1.2 we show an example of the user interface of the receipt transcription task in CrowdFlower used by workers to submit their results.

MicroWorkers (`https://microworkers.com`) – was released in 2009 and by today has around 700 000 workers registered on the platform. The

[2]The author of this thesis went for a 2-months internship at this company in San Francisco in 2013. The internship was financially supported by the program "PhD on the move".

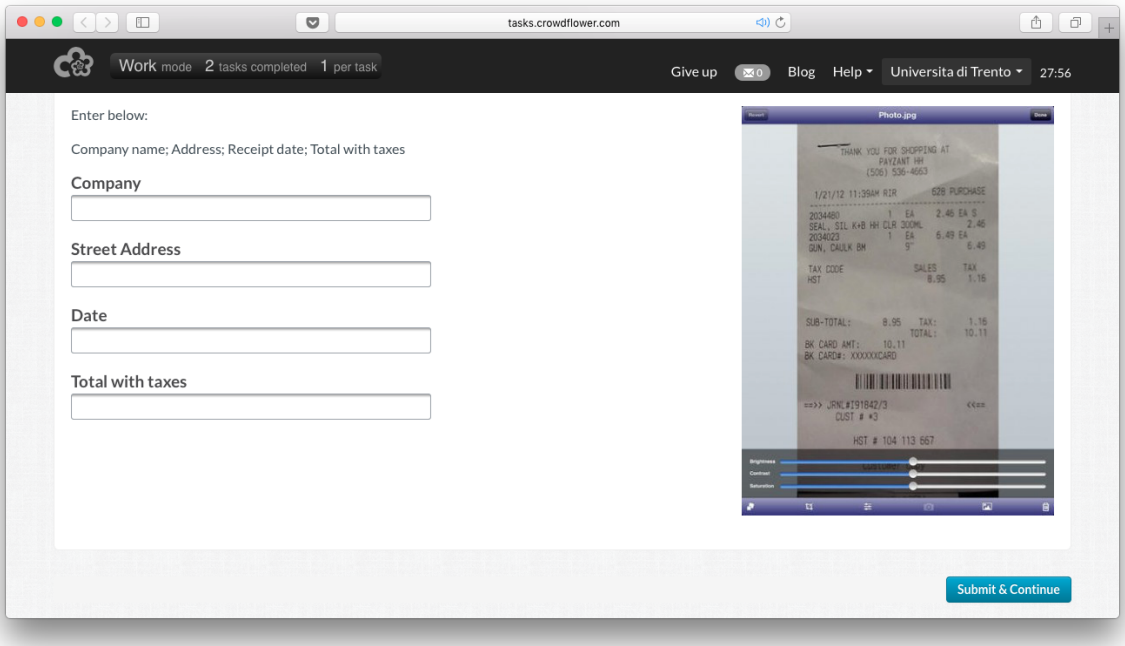

Figure 1.2: The task execution page in CrowdFlower

company claims to have the biggest diversity in its worker base in contrast with MTurk where more than 90% of workers are from the US and India[3]. The platform was extensively flooded with tasks where requesters asked workers to register on their websites to grow their user base or to watch a video to artificially grow its popularity.

CloudFactory (`http://www.cloudfactory.com/`) – was founded in 2010 in Nepal by an American entrepreneur with a social mission in mind to provide work to people from developing countries with fair wages. From its mission the company is similar to MobileWorks we mentioned before. The platform supports generic tasks, still it focuses on 4 use cases, for which specific workflows are designed: text transcription, audio transcription, image labeling and web search. While on MTurk, CrowdFlower and

[3]According to the analysis by Panos Ipeirotis `http://demographics.mturk-tracker.com/#/countries/all`

MicroWorkers anybody can act as a requester and publish a task, CloudFactory works on a contract basis, so only internal employees publish tasks for their customers.

There is a big debate going on about workers motivation and fair compensation in microtask crowdsourcing platforms. Still, according to the report posted by CrowdFlower [93] only about 29% of workers perform tasks to earn money, others do it because they like it (26%) or just want to pass time (28%). This is also in line with the study [56] conducted by Panos Ipeirotis on MTurk, which concludes that 49% of people work for serious income purposes, 42% for entertainment and 34% for some pocket change and 20% to kill time (the total is over 100% because people could select multiple answers in this survey). Learning from workers performing tasks for the sake of income, tells us that consistent workers earn around 300 USD in 20 days working 3-4 hours a day, which seem to be average [30]. Some workers manage to earn close to minimum wages on Amazon Mechanical Turk [41].

1.3 Problems related to quality control

We define quality in microtask crowdsourcing as an ability to meet expectations of requesters [20]. Requesters pay for work and expect it to be done accurately, consistently and in time. Unfortunately because of various reasons it is often not the case. Poor instructions introducing more confusion than clarity, badly designed tasks, unfair reward amounts, anonymous workers performing tasks not paying much attention or even intentionally submitting incorrect results to get rewards faster, programmed bots submitting random results automatically – all these things affect the quality of results dramatically. In microtask crowdsourcing quality control is one of the biggest issues addressed by people both from academia and industry.

2 Survey of Quality Control in Crowdsourcing

We conducted an extensive literature review of papers contributing to different aspects of quality control in crowdsourcing. The paper with this review is to be submitted to ACM Computing Surveys journal [20]. In this section we present a short summary of this paper.

Quality in microtask crowdsourcing is characterized by different dimensions and their attributes, together constructing a *quality model* (Section 2.1). These attributes are evaluated using *assessment techniques* (Section 2.2). *Assurance techniques* help to make sure the attributes are appropriate and of high quality (Section 2.3). We identify specific examples of these quality attributes, assessment and assurance techniques doing a survey of 641 conference and journal papers published since 2009 in certain target venues (the details of the selection are present in the paper [20]).

2.1 Quality attributes

In Figure 1.3 we present quality attributes, examples of which we found in the literature, grouped into dimensions representing the core components of crowdsourcing: input and output *data*, the *task* itself (with its own components: a description, a user interface, terms and conditions and a performance) and people involved (possible actors are: a requester, a worker, a group of workers). Below we discuss how the attributes of these dimensions are covered in the literature.

Data

The high quality of the output data is the ultimate goal of any crowdsourcing task. Data could be characterized by its *accuracy* [42, 61, 128], which could be also called "correctness" [128], "quality" [34, 80, 64] or "goodness" [19]. The level to which responses from different workers for the same data

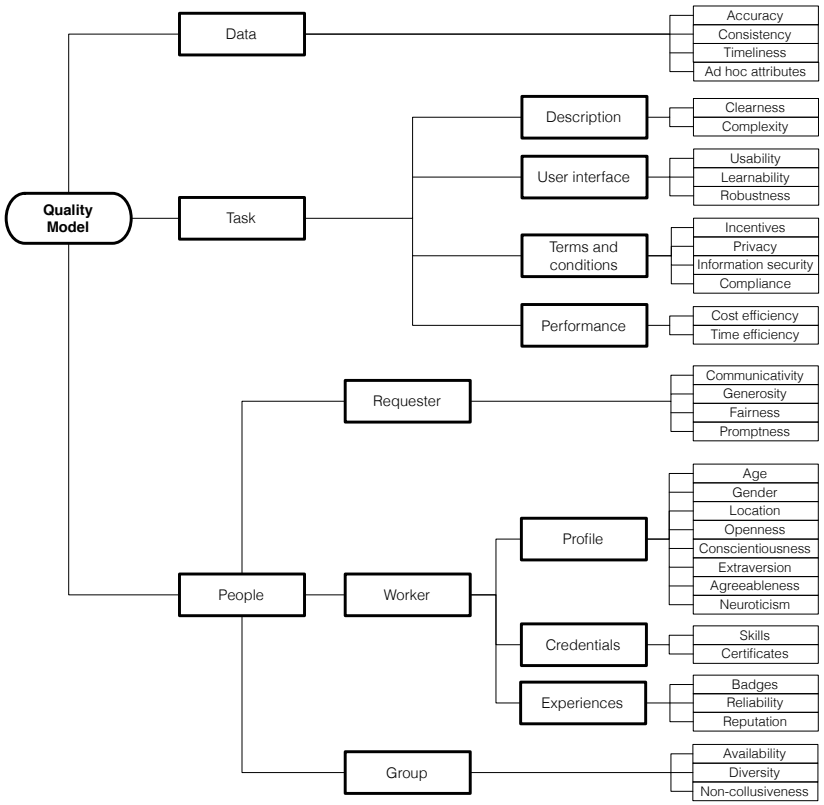

Figure 1.3: Quality attributes (the rectangles on the right) identified in the literature, grouped into quality dimensions (rectangles with bold borders).

input relate to each other is referenced as *consistency* [52, 34]. The *timeliness* attribute represents the probability to get results by a given deadline [70] or as soon as possible [84, 83], which could be referred as "realtime crowdsourcing".

Task

The task description is a set of instructions for a worker to perform the task adequately. Its *clearness* correlates positively [115, 40] with the accuracy of the output data. Workers motivation to work on a task (which further relates to results timeliness) depends on the *complexity* of tasks and their descriptions [50, 82].

If it is clear and straight forward, the user interface can help to involve more workers and potentially increase the accuracy [5]. The *learnability* of the task directly depends on its user interface design [126]. The *robustness* is the property of a task to withstand submissions by adversarial workers [34], which is sometimes referred to as "sensitivity to spammers" [53].

Terms and conditions go beyond a technical implementation of a task. *Incentives* are among the most effective properties, influencing the task attractiveness, resulting in faster execution speed [43, 112]. Incentives could be extrinsic (affecting execution speed) or intrinsic (affecting results accuracy) [48]. *Privacy* is a property dealing with personal data [84]. *Information security* describes how data sensitive to the requester is protected [123]. Whether a given task is in line with certain regulations and laws is characterized by task *compliance* [127], with user policies [125] or ethical requester behavior guidelines [57].

Performance could be characterized by cost and time efficiency. *Cost efficiency* could be counted as the simple sum of costs of individual assignments [8, 90] or in a more precise way be counted as the cost of a single output with high accuracy [55, 103]. *Time efficiency* could be presented as an amount of microtasks performed per unit of time [34]. Kucherbaev et al. improve time efficiency by relaunching tasks during runtime [77].

People

The browser extension Turkopticon [57] enriches the MTurk task list with information about requesters. They are characterized by *communicativity* (how responsive is the requester), *generosity* (how well does the requester pay), *fairness* (how objective and fair does the requester judge work) and *promptness* (how quickly does the requester approve work).

Various worker profile attributes are discussed in [62]. *Age* and *location* do affect the quality of results [62, 34, 63]. There is no evidence that *gender*

has a similar effect. Personality traits are defined in [59] and assessed in [62]: *Openness*, *Conscientiousness*, *Extraversion*, *Agreeableness* and *Neuroticism*. Worker *credentials* are qualifications describing workers experience and background outside of the platform. They could be self-declared or issued by certain institutions. Worker *skills* are intertask abilities, they can be identified automatically [109] and used to match tasks and workers [29]. *Certifications* could be imported on the platform to prove certain skills (e.g. a language certificate) [5]. *Badges* are supported by platforms to indicate that certain actions are performed or goals are achieved [9], as a way to motivate workers [107]. *Reliability* is workers property referring to aggregated accuracy over many tasks performed by the worker [61, 106] or in contrast to an aggregated error rate [23, 26]. *Reputation* is a social attribute indicating as how professional other workers and requesters perceive the given worker [5].

Groups are teams of people working together. The whole crowd could also be considered as a group. *Availability* of workers in general and those with specific skills correlate with accuracy [8] and execution speed [87]. Cultural, demographical and professional *diversity* is specifically important for survey tasks [90]. *Non-collusiveness* is a property of groups of workers, referring to the fact that their members do not disclose and share information with each other and third parties [67].

2.2 Quality assessment techniques

In Figure 1.4 we present assessment techniques grouped into 3 categories according to actors performing the assessment: *individual* – done by an invited expert or by requesters themselves, *group* – when multiple workers from the crowd perform the assessment and *computational* – the assessment is done automatically based on ground truth results or using prediction techniques. Below we discuss the literature concerning these assessment

techniques.

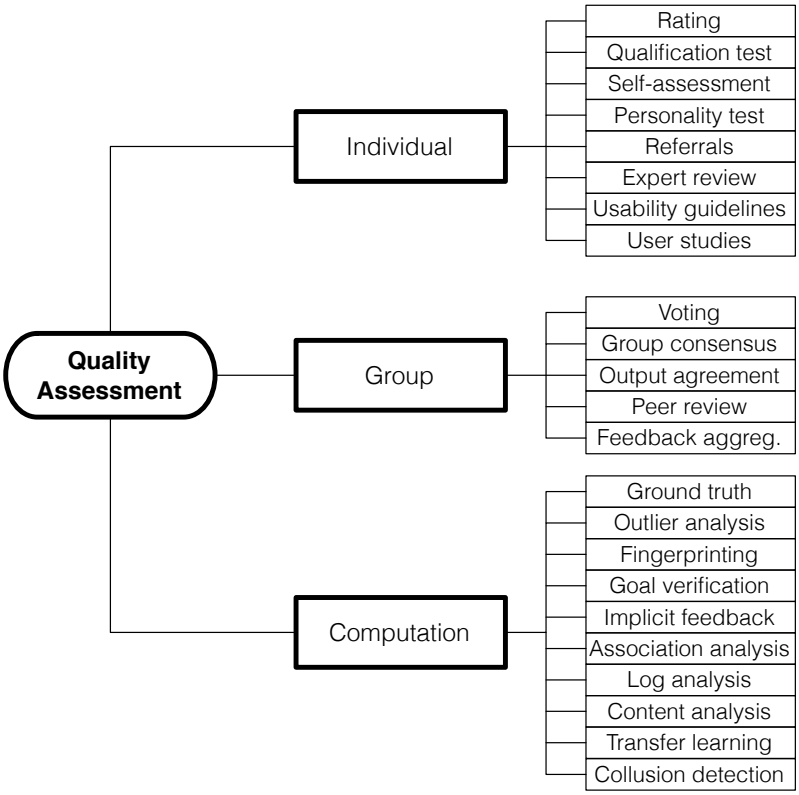

Figure 1.4: Quality assessment techniques (rectangles on the right) identified in the literature grouped by an actor performing the assessment (the rectangles with bold borders).

Individual

Rating is a way to assign a value from a predefined range to a specific result that is under assessment. These values could be such as binary, likert or continuous ones. Rating method is well used for assessing quality of outputs [23, 129], workers confidence [106], task design quality (such as exit surveys in CrowdFlower) and requesters quality [57].

Qualification test is a way to test that workers have required skills. These tests can be evaluated automatically as the correct answers for this test are known in advance. The results of these tests could be reused to as-

sess also other tasks requiring the same test. The study of the effectiveness of such tests shows their benefit to filter out unprepared workers [43].

Self-assessment is a technique asking workers to reflect and perform an evaluation of their own work. Workers in tasks with this technique produce overall better results, than without, also the quality of results of the same workers evolve over time [33]. Asking how confident workers are in their submissions is another way of doing self-assessment [106].

Personality tests aim to assess attitudes and behaviors people possess. This technique can be used to assess workers evaluating their openness, conscientiousness, agreeableness, extroversion, and neuroticism, as discussed in [62, 63], or requesters, assessing their communicativity, generosity, fairness, and promptness, as discussed in [57].

Referrals are recommendations of people who fit best for some task. Bozzon et al. [17] discuss how workers could be recruited from social networks, such as Twitter, LinkedIn and Facebook.

Expert review is performed by a person with deep knowledge in the domain. This person could be the requester himself or a person directly assigned by the requester. An expert can also provide feedback to workers during runtime as in Shepherd system [33].

Usability guidelines originally created as recommendations for developing easy to use applications could also be applied as a check-list for assessing existing user interfaces [95]. Willett et al. in [126] came up with seven guidelines designed 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.

User studies could be conducted to observe workers behavior during task execution in order to understand weak aspects in task user interfaces and to come up with possible ways of improvement [65]. Alagarai et al.

[4] used eye tracking to understand the cognitive demand differences in different task designs.

Group

A group of workers can *vote* for the best fit among result options according to voting instructions. Voting is used to make group decisions, such as in Turkomatic [80], where workers vote for the most appropriate answers. In Turkit [89] voting is supported as a specific task type. Caragiannis et al. [21] discuss various ways of voting in crowdsourcing. Sun et al. [113] discuss some drawbacks of this approach that need to be considered when it is applied.

Group consensus has some similarities both with voting and rating. The idea is not to identify the best result, but the most popular one. Sheshadri et al. [111] compare different approaches to identify relative labels in a collection of labels with a lot of noise. Eickhoff et al.[34] identify workers submitting results without paying attention to the task itself by measuring disagreement with consensus results.

Output agreement shows if for the same task several workers came up with the same or a similar result. Agreement is used to evaluate workers reliability [124].

Peer review is similar to an expert review, utilizing multiple workers from the crowd. Zhu et al. [132] discuss how this method can help to improve results given by workers who acted as reviewers in previous tasks. In some contexts peer review could be more effective than having an arbitrator solving disagreements among volunteers [42].

Feedback aggregation is a way to post-process results to make them less subjective. Similar to how product reviews are averaged among all reviews and how a decision is made in peer reviewed conferences, based on weighted ratings according to the expertise of researchers. Allahbakhsh et

al. [7] weight pairwise evaluations among community members using time and credit of tasks. Other aggregation algorithms exist [54], some with confidence intervals generation for aggregated values [58].

Computation

Ground truth data (the one for which correct responses are already known) could be injected into the overall dataset to automatically filter out workers who do not perform well. While this method is considered as an objective mechanism to measure performance [52], it introduces an extra cost and time for the requester to generate these ground truth data. At CrowdFlower Oleson et al. proposed a way to use results given by trusted workers as ground truth for other workers in their approach "programmatic gold" [96]. Le et al. identify that uniform distribution of ground truth questions produces better results than other types of distribution [86]. CAPTCHA[4] is an example of the ground truth approach to identify if the result is coming from a human or a machine [85, 122].

Outlier analysis allows to identify results or behaviors significantly different from others [3]. Such outliers could be considered as poorly performing workers (e.g. spending too little time). Rzeszotarski et al. [104] discuss how poor performers could be identified visually using graphs generated by the CrowdScape tool.

Fingerprinting is an analysis of workers behavior on the task page to further predict the accuracy of submissions. The term itself was introduced by Rzeszotarski et al. along with a set of accuracy predicting features [105].

Goal verification is an approach to validate if predefined goals are achieved by workers so they get a badge, an internal certificate or another proof of achievement. We believe that this approach refers to Scouts movement where badges could be collected in a defined sequence [98]. Badges are

[4]Completely Automated Public Turing test to tell Computers and Humans Apart

further used as a motivational instrument.

Implicit feedback is a way to extract feedback by analyzing behavior of evaluators [25], rather than in a direct way (e.g. running surveys). WikiTrust [1, 2] is a reputation management tool, where the reputation of a given user is evaluated based on if following workers keep or remove the changes made. Difallah et al. [29] recommend tasks to workers based on personal preferences from social networks (e.g. represented as likes).

Association analysis weights nodes according to the nodes they are connected to. This approach is used in LinkedIn to show the connectivity to unknown professionals and to recommend friends in Facebook. Rajasekharan et al. proposed an algorithm based on Page Rank to compute the "community activity rank" to assess workers reputation [100].

Log analysis is used to make decisions based on the information logged during the execution. Kucherbaev et al. in [77] use linear regressions to estimate the longest assignment duration and consider it as the duration limit for this task. The assignments taking longer are considered as abandoned and are given to other workers to speed up execution. In Turkalytics, Heymann et al. [45] provide real-time information, such as workers demographics.

Content analysis is a way to automatically assess task user interface, task description or task results. Artz et al. discuss "common sense" rules [12], where for example prices below 50% of the average price are discarded. Difallah et al. [29] use content analysis to assess task difficulty. Alagarai et al. [4] analyze input field labels and conclude that too diverse labels might generate distractions and therefore lead to poor accuracy.

Transfer learning is an approach of knowledge transferring from a relevant task [116]. Fang et al. [37] use this approach to estimate workers expertise for data labeling tasks on MTurk. Zhao et al. [131] apply this approach to get knowledge about workers from Yahoo! Answers.

Collusion detection is a way to identify colluding workers such as those who collude about the responses to provide to certain tasks to trick agreement check algorithms. Ground truth data is a way to identify colluders [92]. Allahbakhsh et al. [6] compute collusion probability by analyzing workers who performed the same tasks previously and to recommend new combinations of workers which are less likely to make collusions.

2.3 Quality assurance techniques

In Figure 1.5 we summarize quality assurance techniques according to strategies they utilize: *improve data quality*, *select people*, *extrinsic motivation*, *intrinsic motivation* , *training workers*, *task design improvement* and *execution control*. Below we discuss how these assurance techniques are covered in the literature.

Improve data quality

Cleansing input data is an approach focusing on improving the input data, as this is a precondition for the good quality of the output data [66]. Sometimes workers might even not accept to work on tasks having low quality of input tasks (e.g. image is not loading or is too small and blurry in an image labeling tasks), as they might believe their work could be considered of a low quality and not rewarded [110]. Bozzon et al. [17] propose data pre-processing operations to maintain data in good condition in CrowdSearcher.

Aggregating outputs leverages the "wisdom of crowds" phenomena described earlier [114] to get less biased results from a collection of subjective opinions. Still this kind of technique implies extra costs caused by redundancy in responses collected. Similar to peer reviewed conferences, responses of more experienced workers could be weighted more [13].

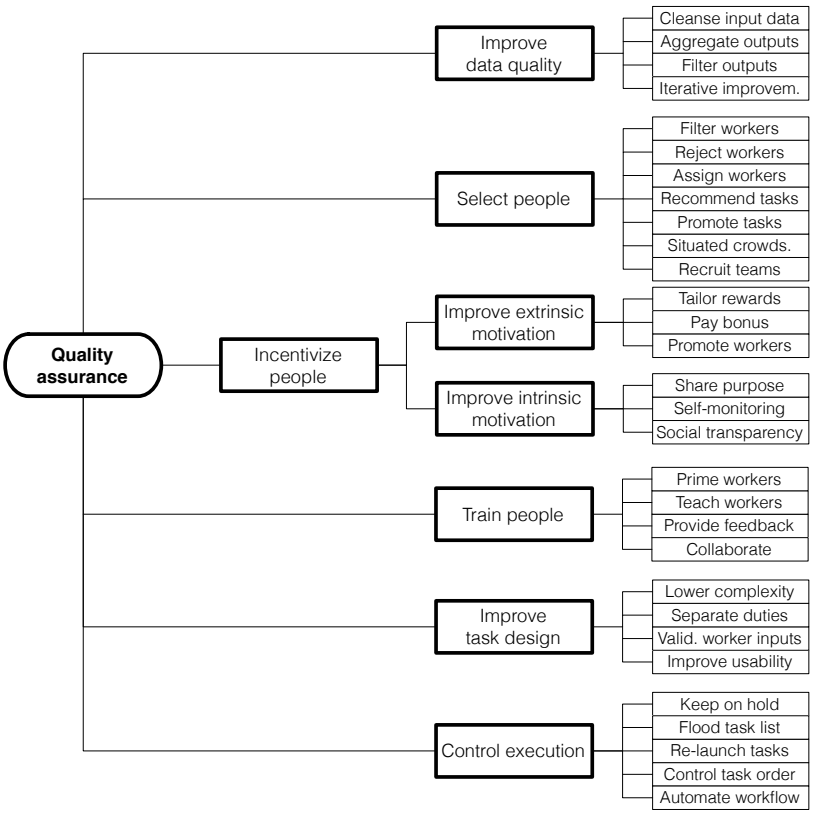

Figure 1.5: Quality assurance techniques identified in the literature grouped by strategies (the rectangles with bold borders) utilized.

Filtering outputs is a natural approach of discarding results which are assessed to be of low quality, such as based on expert reviews [33], output agreements, peer review [42], ground truth [92] or past performance [60].

Iterative improvement is a technique, where instead of doing an evaluation of results, a worker is given a task to improve them. Little et al. apply iterative improvement for writing tasks (e.g. image description) [89] and text transcribing tasks [88].

Select people

Filtering workers is a method similar to filtering outputs, but addresses workers. Filtering can be based on attributes of worker profiles [63], skills,

expertises [5, 131], badges (CrowdFlower), personality [63], reliability [87] and reputation [7].

Rejecting workers ensures that workers who already submitted results of low quality are discarded from a given task or even all the following tasks of the requester. The rejections could be made based on one assessment technique.

Assigning workers is a proactive approach to select workers to perform the task, rather than to wait workers to select it. Workers could be assigned based on time priorities [79], skills and experience [5]. It is also possible to assign tasks to workers external to the platform, such as when experts are recruited from social networks [81].

Recommending tasks is an approach similar to assigning workers, but here workers can decide whether they want to work on the task or not. These recommendations could be made via en email [14] based on subscriptions created on `www.turkalert.com`. Yuen et al. propose to recommend tasks based on worker's browsing history (similar to the way products are suggested in Amazon) [130].

Promoting tasks is a way to attract more workers to the task and could be even done outside the platform. Hu et al. develop widgets with tasks to be integrated on third party websites [50]. Specific forums (e.g. `http://turkernation.com`) and Reddit pages (e.g. "HITsWorthTurkingFor") are also used for this purposes.

Situated crowdsourcing aims to attract workers to perform tasks in a physical world, such as using kiosks placed at a library entrance [47] and paying for snack by performing tasks in special vending machines installed in a hall of a computer science department [44].

Recruiting teams addresses the problem of attracting groups of workers with necessary skills to perform the task. Trial tasks help to identify workers performing well [87] or even experts [101] to target them in the future

with similar tasks.

Extrinsic motivation

Tailoring rewards is an approach of defining and adjusting a reward – the key property of a task. Mao et al. study the task performance with different rewarding schemes, such as volunteering, pay per time, pay per task, pay per each data unit [91], showing that workers behave differently. Researchers propose different motivation strategies: deferred compensation [107], performance based [106], maximization of units completed under a given budget [112] and gambling-based [103].

Paying bonus is an approach of granting exceptional performance, an achievement of predefined goals [107, 27], fast reaction time [128] or additional tasks performed in a sequence [129].

Promoting workers means providing higher positions than ones workers already have. Promotion usually leads to higher rewards. Structures in crowdsourcing platforms are flat and no lower/higher positions are available. Dow et al. [33] promote workers in their tasks from content producers to assessors. Scekic et al. experiment with notions of demotion and punishment [108] in crowdsourcing.

Intrinsic motivation

Sharing purpose is an approach, to attract workers who primarily work for the overall goal of the task, rather than for an extrinsic reward, potentially in a format of volunteering for free [31]. Such, people on Zoouniverse perform tasks to feel connected to a scientific community [31]. Volunteering work attracts less adversarial workers as it has no monetary reward.

Self-monitoring enables workers to compare their performance with others on the platform, which leverages humans desire to compete, pushing them to provide a higher quality of results [55, 107]. This approach can

take different forms: individual performance, overall crowd performance [55] and leaderboards [103, 31, 97].

Social transparency means that workers share their identity and performance indicators with other workers [51], therefore creating personal connections assuming some level of personal responsibility, leading to better performance [51, 121]. Yu et al. suggest that positive relationships among workers makes the workplace (e.g. the task or the platform) more attractive for other workers [129].

Training workers

Priming workers aims to bias workers without them consciously understanding that their behavior has been changed [94]. Different types of content, such as images, text, audio and video can evoke various emotions in workers positively affecting the quality of the outputs [94, 36]. Alagarai et al. use this approach to let workers remember information better [4].

Teaching workers is a way to give new and improve existing skills of workers, which is already a motivation for some of them. The teaching could be done in-person as in Samasource or via tutorials [31] in Mobileworks. According to Yu et al. designing tasks helping workers to obtain skills results in better performance [129].

Providing feedback to workers about their performance helps to obtain better task results [33]. In Turkomatic requesters' feedback to workers helps to solve complex tasks better [80]. According to Yu et al. [129] encouraging workers to review each others outputs improves workers skills and leads to better performance.

Collaboration happens when several workers perform the same task and a negotiation [68] takes place, a workplace [11] or data is shared. Dorn et al. introduce workflows to organize workers into collaborative teams [32].

Task design improvement

Lowering complexity of tasks helps to improve the quality of results [102]. It is important to design tasks cognitively in the simplest possible way, as for example comparing a pair of objects is easier than identifying specific features of individual objects [10]. Turkomatic platform is based on lowering tasks complexity, done by workers arbitrary splitting tasks, following a price-divide-solve algorithm [80]. Kittur et al. introduce CrowdForge, where big work is executed following a partition-map-reduce approach as small individual tasks [71].

Duties separation is an approach coming from the business world, where different people do execution and evaluation so they are not biased and there is no conflict of interests. Bernstein et al. propose a find-fix-verify approach [15], where some workers identify errors, others fix them and later some other verify that there are no more errors left. In Turkomatic workers decide by themselves to perform a task or to split it and let others solve them as subtasks [80].

Input validation helps to automatically do the first check of results, blocking obviously incorrect results, such as a phone number should not include letters or a country zip code should be of a defined limit. In CrowdFlower and AskSheet [99] a requester can define allowed formats for certain fields (e.g. an email, US address). Bragg et al. design and implement tasks where it is not possible to complete an assignment unless at least one option of multiple choice field is selected [99].

Improve usability aims to make the crowdsourcing task user interface less cluttered and more convenient for workers so they can perform better [61]. One good practice is to show to workers clear instructions along with examples of good work [126]. CrowdFlower recently adopted this practice for task design. Another good practice is to have question labels close

to input fields where answers should be entered [4]. Sometimes it is also possible to design tasks in a way that to perform it well takes the same or less time than to cheat, leading to better quality of the results [69].

Execution control

Keep on hold is an approach to minimize the reaction time when new tasks are submitted on the platform, here workers are also paid for waiting new tasks [14]. To assure that workers are focused they could be given simple cognitive tasks or even games during waiting for real tasks [84]. Because it could be expensive to pay to a pool of workers to stay active, several requesters could share a single pool for their tasks [16].

According to Chilton et al. [22] workers select primarily newer published tasks, rather than farther pages of the task listing page. Bernstein et al. propose a *flood task list* method when tasks are repeatedly posted to keep them on the top of the task listing page and therefore have more workers involved [14].

Relaunching tasks is a method we introduced [77] to monitor task execution during runtime, cancel problematic ones (which considered as abandoned and could delay the overall execution) and launch the same data units, so other workers can execute them straight away. Bozzon et al. propose workflow adjustments during runtime [18].

Task order control in some tasks could lower overall costs by not deploying some microtasks, according to results given for other microtasks. Such as in a comparison task if $a = b$ and $b \neq c$, there is not reason to compare a and c [120]. Marcus et al. [92] discuss how to apply knowledge from database query optimization to make the sequence of tasks ordering more efficient.

Workflow automation is a way to approach complex work, which could not be presented as a single multiple instance task, but requires a workflow

of multiple tasks executed sequentially and in parallel. In [76] we provide an overview of different research and industrial approaches to automate workflows.

2.4 Open challenges and opportunities

In our survey [20] we analyzed 15 different crowdsourcing platforms based on our evaluation framework. From this analysis we understood which quality attributes, assessment and assurance techniques are covered well and which poorly. Here below we discuss poorly supported attributes and techniques as opportunities for future research tracks and industry products.

Quality Attributes

Personality We believe it could be positive apart from hard skills to introduce soft skills based on worker personalities on crowdsourcing platforms. In such an environment workers could perform better, collaborate with others and requesters more productively, as they feel comfortable at their virtual workspace.

Transparency In 2004 with the appearance of Facebook social networks converted from communities of mostly anonymous people to communities of people with real identities. This was a big step and resulted in having social networks with 1 BLN user bases. We believe that real identities could improve the level of trust in crowdsourcing platforms, while now, not only workers but also requesters are mostly anonymous, decreasing the level of personal responsibility.

Group work Even though there are some attempts to introduce teams into crowdsourcing platforms, this domain is still not well understood. Further

research about ways to organize groups, identify team members, groups structures, hiring and other aspects of teamwork are necessary.

User interface quality While it is confirmed that good task user interfaces result in higher quality of the outputs, there is no much support for requesters to develop high quality task forms apart from individual best practice articles.

Quality Assessment

Self-assessment We believe that self-assessment is a technique which is very effective and easier and cheaper to implement than ground truth data, however, it is still almost not adopted at all in crowdsourcing platforms.

User-interface assessment Apart from ways to develop high quality task user interfaces it is important to have approaches and tools to evaluate them, and generate a set of suggestions for the requester for improvements. An automatic assessment of user interfaces could help workers to filter tasks only with very good interfaces, enforcing all requesters to pay attention to this aspect of their tasks.

Runtime assessment An automatic evaluation of tasks, such as those based on ground truth data is usually used after a task is completed. Doing assessment during runtime could help to adjust task execution to get better accuracy of results and faster overall execution time.

Quality Assurance

Task recommendations On platforms, such as MTurk, where at a given moment there are thousands of tasks available, it is crucial to have better ways

to navigate across tasks then simple filtering and sorting techniques. Recommendation of similar tasks (as products in Amazon) along with stream generation of similar type of tasks (as radio-stations in Spotify) could be of use.

Long-term relationships To let workers stay involved in performing tasks in a long term, tighter relationships should be built. Learning skills helping to perform new types of tasks and to assure a certain level of income could be a possible solution.

Workflow automation Nowadays it is hard to find simple atomic tasks, which are not parts of bigger work. It is important to have approaches and tools to design and deploy complex crowdsourcing tasks in a user-friendly way, rather than manually programming execution logic for every complex task.

3 Research Tracks

During the time of this PhD we did research in the following topics identified in the prior section: *user interface quality*, *runtime assessment*, *task recommendations* and workflow automation. We carried our work in 3 research tracks, as is shown in Figure 1.6. All research tracks contribute to the common topic of improving quality in microtask crowdsourcing. We consider *Improving accuracy and speed* (using runtime assessment) as the primary track and during the defense we focus on it the most. Two other satellite tracks are *Crowdsourcing complex processes* (using workflow automation) and *Improving workers experience* (using task recommendations and focusing on user interface quality). In Figure 1.6 the rectangular blocks are publications associated with corresponding tracks. We color

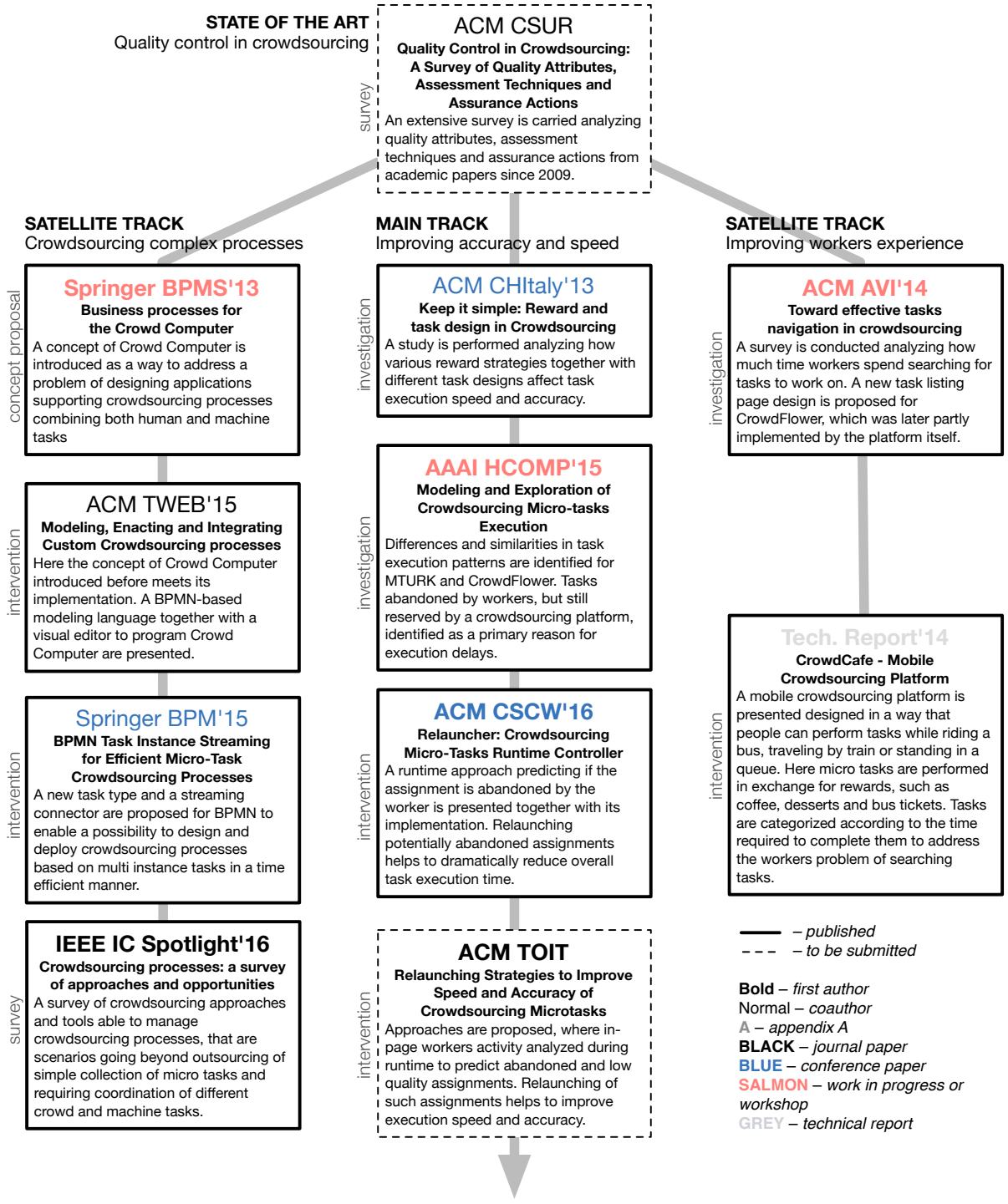

Figure 1.6: Three research tracks were carried, all contributing to the topic of quality control in microtask crowdsourcing: *Crowdsourcing complex processes*, *Improving workers experience* and the main track – *Improving accuracy and execution speed*.

code publication types into: journals – black, conferences – blue, workshop or works in progress – salmon and technical reports – gray. Contribution types (survey, investigation, concept proposal, intervention) are indicated on the left side of each rectangle. The author of this thesis is the first author in publications where venue titles are written in bold font. In total there are 11 papers: 2 published journal papers [118, 76], 2 journal papers to be submitted [20, 73], 3 published conference papers [38, 77, 117] and 3 published poster and workshop papers [75, 78, 74]. We also include one technical report [72].

3.1 Crowdsourcing complex processes

Microtask crowdsourcing by its nature is designed to deal with atomic small tasks. Still these small tasks are usually only parts of bigger and more complex work. Dividing the complex work into a collection of interdependent microtasks could be a not trivial job. It still requires some manual data processing, manual deployment of each underlying microtask, data passing management among these microtasks, according to an underlying business process. We call such complex work, requiring integration of various tasks, performed by machines, crowd workers and individual people – *crowdsourcing processes*.

Our approach: BPMN-based Crowd Computer

To address the problem of crowdsourcing complex processes we introduced a concept of *crowd computer*. There are two computing components in crowd computer: a traditional computer (a machine) and a crowd. The *crowdsourcing engine* controls the execution of instructions set utilizing both these components. We argued that the most appropriate way to design instructions is in a form of a business process, and a notation, such as BPMN, could be a good instrument for that. Such crowd computer has a

storage unit, which besides data and instructions, stores information about crowd workers. *Crowd interaction component* connects the crowd with the crowdsourcing engine via a graphical interface, through which workers perform instructions. A developer designing crowdsourcing processes for such crowd computer utilizes a repository of crowdsourcing process templates. These templates include general *tactics* how to approach crowd (e.g. a marketplace, a contest or similar) and specific *operations* how to preselect workers, do quality assurance, aggregate worker results and other.

We published and presented our concept proposal at BPMS workshop in 2013 [78].

Later, based on the introduced concept we implemented Crowd Computer[5] with a proprietary platform for performing crowd tasks and web-service calls support to perform machine tasks. In order to have real workers executing tasks we implemented an integration with MTurk to publish crowd tasks also there. We introduced BPMN extensions to program Crowd Computer together with a visual interface (Eclipse plugin) to design business models using this modeling notation.

This work was published in ACM Transactions on the Web journal in 2015 [118].

The multi-instance tasks included into BPMN are not designed to allow *streaming* – such when an instance of one task is finished, the associated instance of the following multi-instance task starts. Without streaming support running two sequential tasks, such as image tagging and image categorization is time inefficient, because first all images should be tagged in the first task and only then image categorization task starts. To solve this problem we proposed a new task type (crowd task) and a streaming connector as an extension to BPMN. These extensions allow to design and deploy complex crowdsourcing processes supporting streaming, and the

[5] `http://www.crowdcomputer.org`

introduced connector manages splitting and merging of data to pass to following tasks.

This work was published and presented at ACM BPM conference in 2015 [117].

Survey of other approaches

Further we did a survey of other approaches allowing to crowdsource complex processes. We identified 11 approaches about which we could find ether enough information in a form of an academic publication, or if an implementation of this approach was available to be tried out. These 11 approaches were grouped according to the paradigm of their process definition language:

- *Imperative, textual* – the requester defines process in a form of a programming code (e.g. using JavaScript or Scala),
- *Imperative, visual* – the requester visually models how to execute the process (e.g. using BPMN as in case of Crowd Computer),
- *Declarative* – the requester defines what output should be obtained (e.g. using SQL or spreadsheet formulas),
- *Configuration* – the requester configures the process as special case of a given generic process (e.g. following a wizard-style).

In this survey we analyzed that while currently a set of approaches already exists or proposed, they stay more as research prototypes and are not well presented in the industry. Even CrowdFlower and Workfusion have similar tools, still they are ether used only internally by the employees or are only available to top-tier customers. As a result of this survey we proposed a set of directions to advance solutions, which we discussed together with Lukas Biewald, CEO of CrowdFlower:

- *Integration* – proprietary notations for process definition might make it more complex to integrate with other programming environments.
- *Quality control* – as this is a primary topic in microtask crowdsourcing, having a built-in extensible module for controlling quality is a must,
- *Adaptive process execution* – a convenient way for testing processes before running them with big datasets is required, adaptations during runtime is an option,
- *Worker selection and training* – to address a problem of identifying workers with required skills and if there are none, training workers.

This survey was published as a spotlight article in IEEE Internet Computing magazine in 2016 [76].

3.2 Improving accuracy and execution speed

The track about accuracy and task execution speed improvement started opportunistically after following "Computer Supported Cooperative Work" course given by Gregorio Convertino at the University of Trento in 2012.

Reward schemes and task user interfaces

We investigated how different task design and reward schemes affect results accuracy and speed. For that we ran two experiments on Amazon Mechanical Turk. In each experiment we asked workers to transcribe a handwritten text.

In the first experiment we analyzed the influence of different reward schemes applied independently and together to accuracy and speed (conditions: no motivation, "please do it well", "if you do it well you get a fixed bonus", "depending how well you do you get variable extra bonus"). This experiment revealed an interesting fact that simply asking workers

to perform the task faster brings a noticeable speed effect. Homogeneous schemes (those that apply the same logic to both accuracy and speed) showed the best performance speed wise. We believe the reason behind it is the lower cognitive demand, as these kind of schemes are much easier to understand.

In the second experiment we analyzed, how different user interfaces (a cluttered and a clean one) affect accuracy and speed. On top of a regular transcription task in one condition we gave a cognitive task (to recall a shape shown on a previous task page). The results of this experiment are relatively trivial still are important: "keep it simple" – we got responses of higher accuracy in conditions where extra cognitive task was not introduced and where the user interface was clean.

This work was published [38] and presented at CHItaly conference in 2013.

Understanding task execution process

Later when we worked on task streaming [117] in our experiments on CrowdFlower we noticed performance peaks every 30 minutes (Figure 1.7) we did not know the reason for. In order to investigate and identify the underlying reason we conducted two experiments.

In the first experiment we ran a receipt transcription task on both CrowdFlower and MTurk with 3 different conditions: i) reward is 0.10 USD, no preselection; ii) reward is 0.10 USD, only skilled workers are allowed; iii) reward is 0.01 USD, no preselection.

We identified that on CrowdFlower there execution parallelism is stronger (Figure 1.8), which means that there are more workers involved in the task at a given time. Because in CrowdFlower the demand-supply ratio is higher (more workers, less tasks), workers started performing the task already in the first 10 seconds since the task publication time. On MTurk the par-

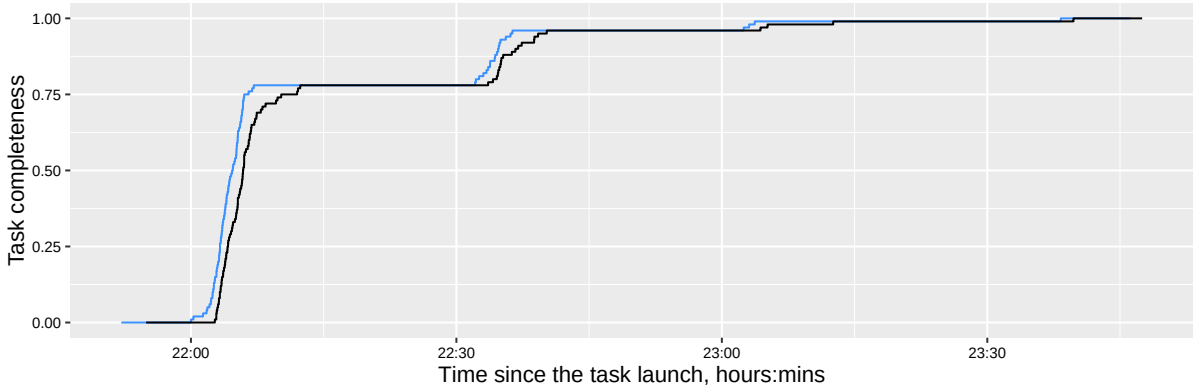

Figure 1.7: Cumulative task start (blue) and completion (black). There are performance peaks approximately every 30 minutes.

allelism is lower. Another interesting observation is that on MTurk some workers open several instances of the same task in different browser tabs (worker "UU42" in MTurk condition 2 in Figure 1.8) in order to reserve more work for themselves. Such behavior is not technically possible in CrowdFlower, as there workers can only have single assignments in one task. The reason for 30 minutes productivity peak intervals happened to be the maximum execution time alloted by CrowdFlower. If a worker there does not finish the assignment and simply closes the page tab without properly leaving it clicking the corresponding button (called "give up"), the platform keeps this assignment reserved for this worker for 30 minutes. In such a way when several workers start their assignments, but later abandon them, the assignments will be released and become available for other workers only in 30 minutes.

In the second experiment we ran the same task on CrowdFlower only varying the reward amount (from 0.01 USD to 0.25 USD with the step = 0.03 USD, the number of units = 20) and the number of units in the task (from 10 to 100 with the step = 10, the reward = 0.01 USD), having 18 conditions in total. To our surprise we identified that higher rewards do not cause shorter assignment start times (probably because they are

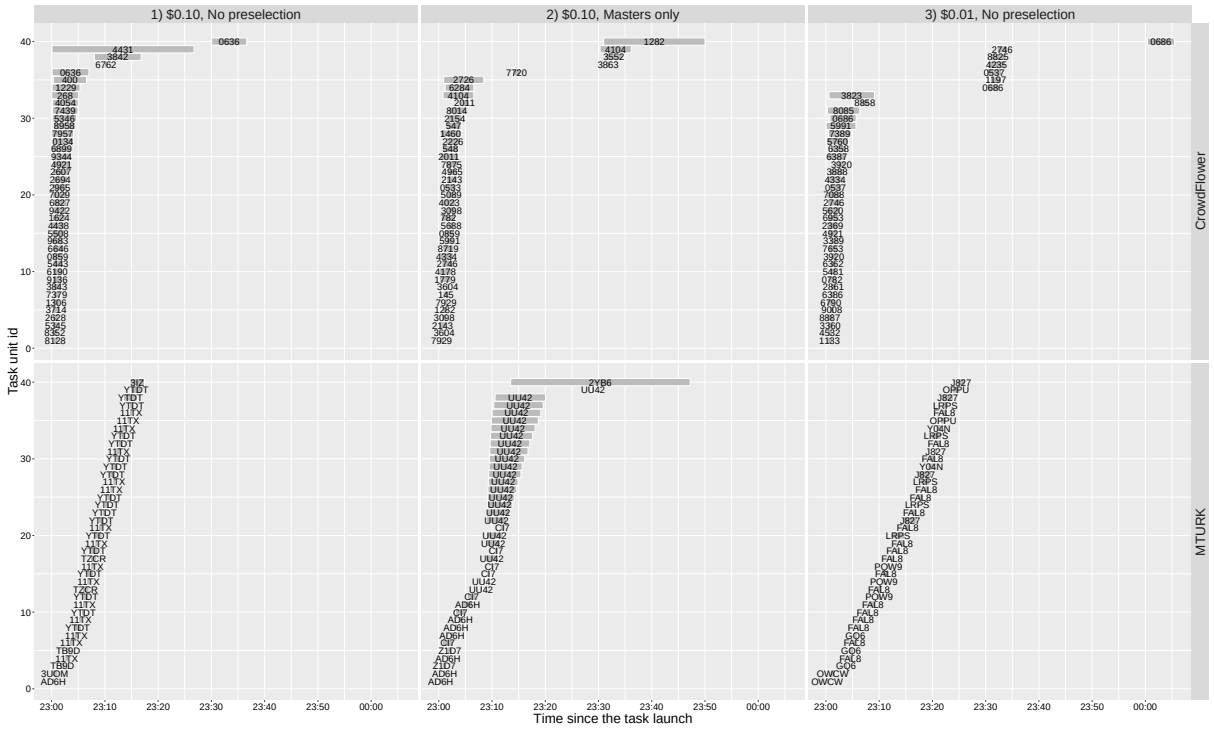

Figure 1.8: The execution timeline of the receipt transcription task ran on CrowdFlower and MTurk in three different conditions. Gray bars correspond to assignments. Codes on top of bars correspond to worker identifiers.

already very short) or assignment durations. Gadiraju et. al identified [39] that in CrowdFlower "easy to complete" task property is even more important than reward, which could be the reason for our finding.

This investigation was presented as a poster [75] at AAAI Human Computation conference in 2015, which is the key conference in domain of microtask crowdsourcing.

ReLauncher The life cycle of a data unit is presented in Figure 1.9 as a colored Petri Net. Originally all data units are in "to be assigned" state. When workers select the task, the crowdsourcing platform creates assignments connecting the units and the workers, moving the data units to the "started" state. From the "started" state these data units can be

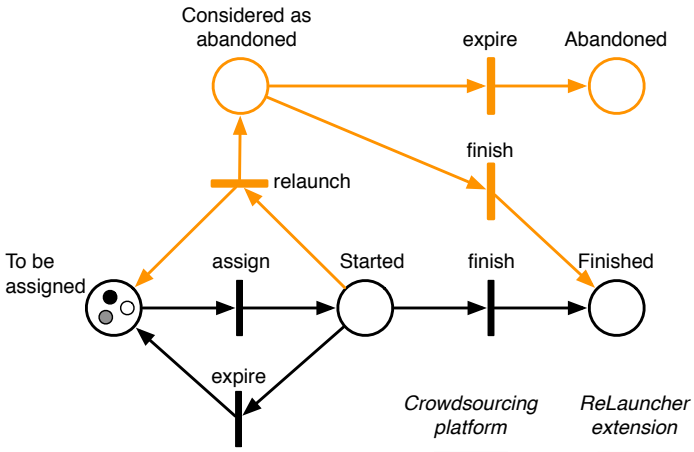

Figure 1.9: Colored Petri Net of a data unit life cycle. Black – the crowdsourcing platform, yellow – ReLauncher extension.

finished by workers or expire if they are abandoned. If abandoned the units go back to "to be assigned" state and other workers can join them.

We propose an extension on top of a crowdsourcing platform which monitors units in "started" state and predicts if the units are abandoned or not. If predicted as abandoned they are *relaunched* (which mimics the expiration procedure performed by the platform) - the copy of the unit is made and launched on the platform again, while the original data unit is canceled. If the prediction is correct then the assignment in the original data unit gets expired and the unit moves to "abandoned" state, otherwise it is finished and the requester pays for 2 assignments for a single data unit. On CrowdFlower assignment start times are in average very short. Because of that, as it is shown in Figure 1.10, the results for assignments with shorter durations tend to arrive faster (have lower completion order index) than for assignments with longer durations. In such a way the result for the longest assignment could be assumed be the latest to arrive. There is a correlation between assignment durations and assignment completion order. We calculated a linear regression model for

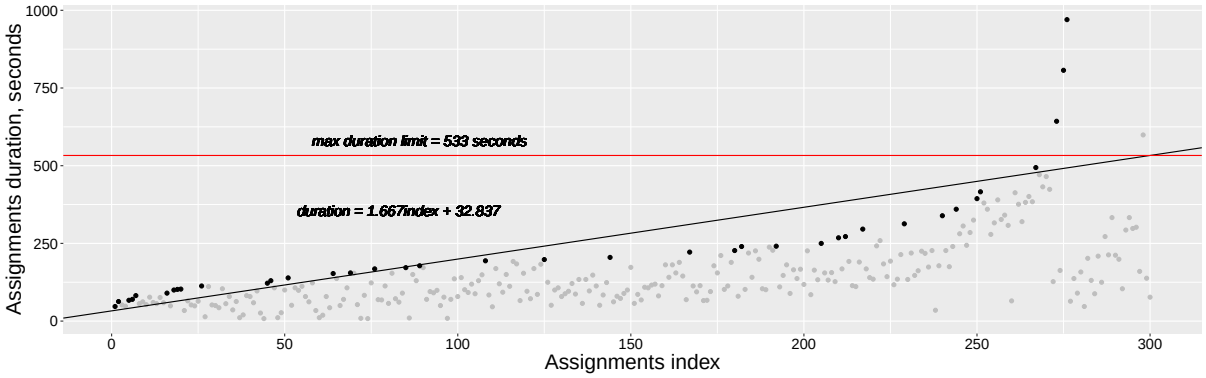

Figure 1.10: The linear regression approach to predict abandoned assignments.

local maximums – assignments having the longest durations among those completed already before (black dots in Figure 1.10). With this model we predict the duration of the last assignment, use it as a time limit and consider the assignments staying in "started" state for longer than this time limit as abandoned. We calculate this limit during runtime, cancel and relaunch units of assignments which we consider as abandoned. Even if our prediction is incorrect (false positive), a worker still can finish the assignment and get a reward, so our solution does not affect workers in any way.

We implemented this approach as a web tool called ReLauncher[6] and ran several tasks with and without it. ReLauncher predicts abandoned assignments using linear regression approach gives a recall of 1 (all abandoned assignments are identified), introducing around 10% of false positives (according to our experiment with 5 repetitions). This approach gives several times improvement in the overall execution speed (more than 3 times according to our experiment).

This work was published [77] and presented at ACM CSCW conference in 2016.

[6] `https://github.com/ReLauncher/crowd-relauncher`

Approaches to improve speed and accuracy

To improve the overall execution speed and results accuracy we tried out several approaches to predict abandoned assignments (in addition to the linear regression approach discussed above) and low quality assignments during runtime: *tabs visibility analysis*, *duration outliers analysis*, and *in-page workers behavior analysis*.

In order to evaluate the proposed approaches we implemented a JavaScript file[7] to inject into tasks user interfaces to collect workers activity, such as: browser tabs visibility (e.g. hidden, visible, closed), key presses (which key, when was pressed), mouse clicks (which HTML element, when was clicked), and general activity (if keyboard, scroll or mouse were used). With this JavaScript file injected we run three popular task types: receipt transcription task, image labeling task and business search tasks.

Before to discuss the performance of each individual approach it is important to define what performance we actually aim at. *To improve the execution speed* the approach should identify abandoned assignments with $recall = 1$, as otherwise the delays will still take place. The higher the precision we have the lower is the amount of assignments which were not actually abandoned and for which we pay extra. *To improve the accuracy of results* the approach should identify low quality assignments with a decent precision, which is close to 1. The higher the recall, the higher is the amount of low quality assignments we correctly identify.

Tabs visibility analysis A browser tab of assignment task page can be in *active* (the worker is on the task browser tab), *hidden* (the worker is browsing other tabs in the browser) or *closed* (the worker closed the tab or the browser) status. In the tabs visibility analysis approach we aim to relaunch all data units with assignments for which task browser tabs were closed.

[7] `https://github.com/ReLauncher/worker-activity-logger`

The performance of this approach is presented in Figure 1.11. This approach does not provide better results than the linear regression approach. The reason for that stays in the way CrowdFlower works. Even when workers leave the task by closing the task tab, their assignments stay active and the next time the worker clicks at this task in the task listing page this worker is routed back to the original assignment. This means that each assignment can have several *sessions*. This fact was surprising for us not only technically that it is possible on the platform, but behaviorally, that it is not a rare case that workers first leave and then go back to their original assignments. Because our the tabs visibility analysis approach relaunches assignments every time workers leave their task pages, it introduces too many false positives. Even if the recall is still 1, precision is lower than in the linear regression approach.

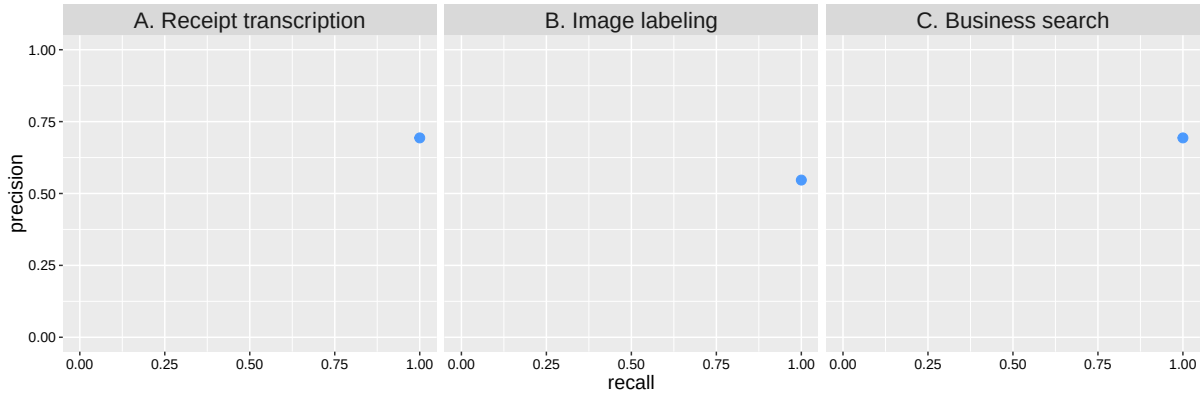

Figure 1.11: The performance of tabs visibility analysis to predict abandoned assignments.

Duration outliers analysis After running several trial tasks we found that for some task types, such as receipt transcription, assignment duration is a good accuracy predictor. In our experiment all assignments with durations of less than 27 seconds happened to have low accuracy (Figure 1.12). Therefore we searched for task independent percentile threshold to automatically cut out assignments having low quality with high confidence. In

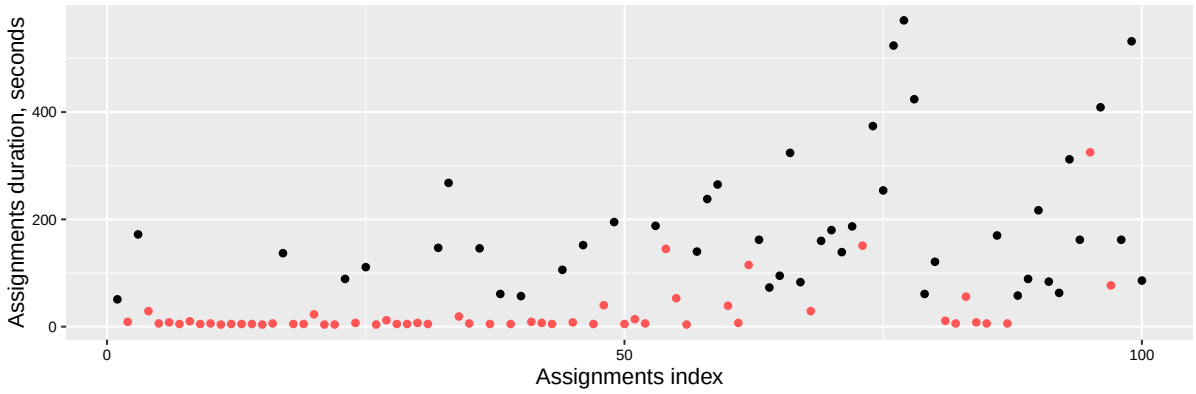

Figure 1.12: Assignment durations vs completion index having low accuracy (red) and high accuracy (black).

Figure 1.13 we show the performance of the approach with different percentiles of assignments considered as low quality. The higher the threshold the higher the recall but lower the precision. We identified that in the receipt transcription task cutting out the fastest 4% helps to remove half of the low quality assignments with a precision close to 1. The approach does not work for the image labeling task, because here the execution duration is not a good accuracy predictor, as typing the relevant or irrelevant word takes about the same amount of time.

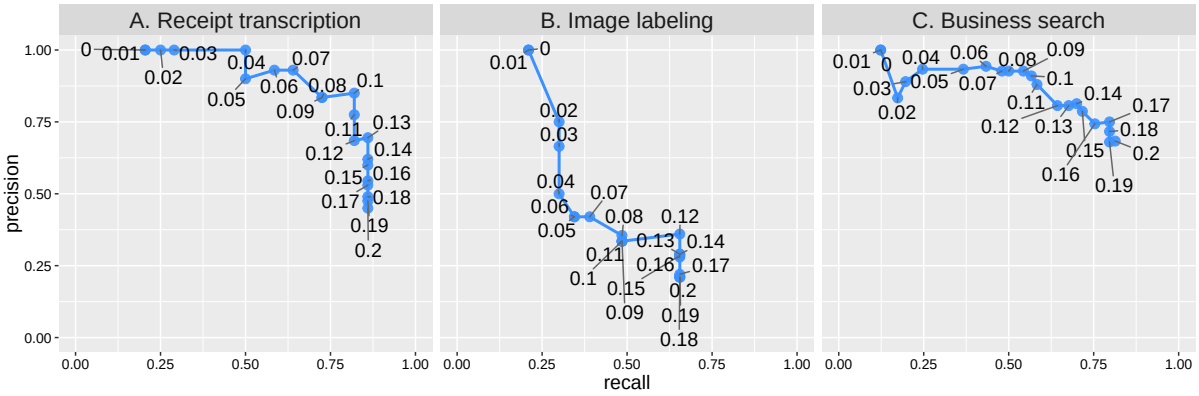

Figure 1.13: The performance of the duration outliers analysis approach to predict low quality assignments. Numbers near dots refer to percentiles of the fastest assignments to consider as low quality.

In-page workers behavior analysis In this approach we derive features from logs of one task run and use them as a training set for another run of the same task type. Using the recursive partitioning method we get a decision tree out of the training set to predict abandoned or low quality assignments.

The performance of the approach to predict abandoned assignments is presented in Figure 1.14. In the tasks we experimented with the following features happened to be good predictors of abandoned assignments: the amount of times any keys were pressed, the assignment duration. The more data we use for the training set the higher becomes the precision and lower the recall. If we use all logs for the receipt transcription task with 100 data units the precision is around 0.85 and recall is around 0.78. While the performance for the image labeling task is similar, the performance for the business search task is poor. Even if precision is pretty high the recall of less than 1 result in still having the problem of execution delays. While this approach was promising it did not show better performance than linear regression approach.

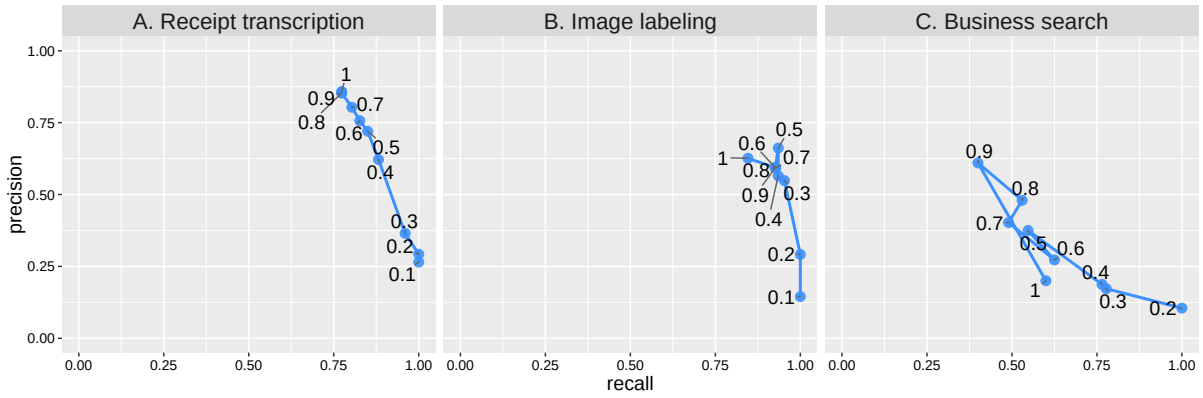

Figure 1.14: The performance of in-page workers behavior analysis to predict abandoned assignments. Numbers near dots refer to the part of the previous task run logs used as a training set.

The performance of the approach to predict low quality assignments is presented in Figure 1.15. In the tasks we experimented with the fol-

lowing features happened to be good predictors for accuracy: the amount of times any character keys were pressed, the delete key was pressed, the punctuation keys were pressed and the assignment duration. For the receipt transcription task, using the logs of the same task ran before helped us to get a precision around 0.9, identifying more than a half of the assignments with low accuracy (recall is greater than 0.5). For the image labeling task no feature we use happened to be a good accuracy predictor and therefore the approach does not bring any benefit for this type of task.

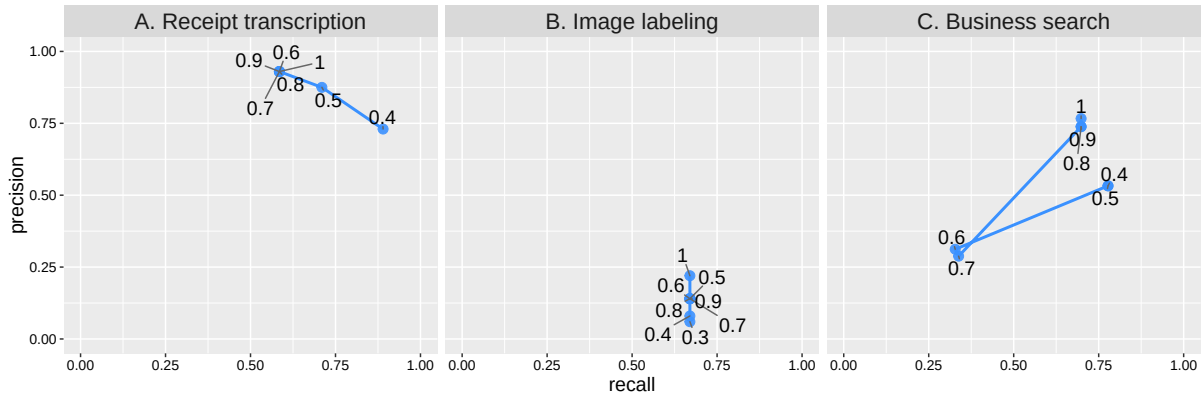

Figure 1.15: The performance of in-page workers behavior analysis to predict low quality assignments. Numbers near dots refer to the part of the previous task run logs used as a training set.

Depending on requester priorities approaches could be combined providing a higher recall or higher precision, such as the linear regression where only assignments with an evidence of workers leaving the assignment (tabs visibility approach) could be relaunched. For the in-page workers behavior analysis more features have to be tried out in order to perform better predictions.

The work discussing different approaches to improve task execution speed and accuracy is to be submitted [73] to ACM Transactions on Internet Technology journal.

3.3 Improving workers experience

Workers are an essential part of microtask crowdsourcing, therefore making sure that they are satisfied and work in a comfortable environment could lead to better results. Still both academic and industry communities stay more requester-oriented, searching ways to get better results with less money and time spent.

Task search

During the internship at CrowdFlower in 2013 we conducted a study on this platform, trying to understand how workers search for tasks to work on and how much time they spend searching. This study revealed that most of the workers spend more than 25% of their time searching for tasks to work on. More than 2/3 of workers reported that they consider searching as a problem and half of those said it is a critical problem. We also asked workers what kind of solution they believe could help to make the searching experience more pleasant and efficient. The most popular answers were "ranked keyword search", where outputs are ordered according to the relevance to workers queries and "suggestion box", which is similar to what Amazon provides for recommending relevant products.

Keeping in mind the results of this study we started to design a prototype of a task listing page for CrowdFlower. In our prototype workers could filter or sort tasks by title, requester name, number of instances inside, aggregated satisfaction level (based on evaluations given by other workers to this task), task category (most of the tasks did not have categories assigned, so we used TF-IDF algorithm [46] to categorize them automatically based on 20000 task titles). This prototype went through a set of iterations with CrowdLab community, which is an online group of trusted workers CrowdFlower employees interact with to get an early feedback.

This work was published [74] and presented at CrowdUI workshop of ACM AVI conference in 2014. Later CrowdFlower redesigned its original task listing page user interface according to our prototype.

Workers oriented platform

With the knowledge coming from the previous study and the overall experience of working at CrowdFlower we came up with an idea to design and implement a microtasking platform CrowdCafe[8], based on the following three principles:

Mobile Workers perform tasks using their smartphones. In such a way people can perform tasks and earn rewards while riding a bus, queuing at a grocery store or waiting for an appointment. We believe that such approach can help bring more people into microtask crowdsourcing, who do not consider it as a primary source of income. On smartphones some actions are more natural, rather than on desktops, such as swiping for action (originally introduced in MailBox[9]), shaking to undo and the voice input. Because microtasks intend to be very simple they can be presented well even on smartphone screens.

Categorization In order to solve the problem of searching tasks discussed earlier we introduced a fixed set of task categories: *espresso* – which takes about 10 seconds to complete; *cappuccino* – taking about 2 minutes to complete and *wine* – taking about 5 minutes to complete. Each task category has a fixed reward amount, so workers do not spend time comparing tasks by their profitability. Requesters are responsible for picking the right category for their tasks. Tasks requiring more time than defined by the

[8] `https://github.com/crowdcafe`
[9] `http://www.mailboxapp.com/`

associated category are reported by workers to the platform.

Motivation Workers do not get cash on this platform as it is not designed as a source of primary income. The idea is to provide tangible rewards, such as a coffee, a bus ticket, an ice-cream or a cellular-network credit. The price of coffee in Italy at a university bar is about 0.60 EUR. It is hard to motivate people to perform tasks for this amount of money, but experiments show that students are very motivated to perform tasks on CrowdCafe for a cup of coffee.

We deployed the platform and invited students at the University of Trento to perform tasks there. More than 400 people registered on the platform and participated in image labeling and natural language processing (NLP) tasks. We identified to our surprise that out of 791 assignments for image labeling task we did not have a single assignment with not relevant labels (we still had responses where labels were in Italian, while we asked for English, and responses having less than 3 labels we originally asked for). We believe that on a smartphone it takes the same time to type something relevant than random, and copy and paste action is not as simple as on a desktop. In the NLP task many people gave random responses and we believe that the reason for that was the low quality of task instructions we provided. We confirmed it in the follow up survey. Around 80% of participants responded they want to use the platform in the future and requested more tasks on the platform.

This work was published as a technical report in 2014 [72].

4 Summary of Contributions

We summarize contributions in four groups: i) *analysis of state of the art*, where we conduct the most extensive survey on quality control in

crowdsourcing so far; ii) *empirical studies*, which are based on a set of experiments we conducted with a purpose to understand better the domain of crowdsourcing and specifically the crowd behavior; iii) *interventions*, which are the approaches we designed, implemented and tested, trying to improve certain quality dimensions; iv) *software prototypes*, with which we refer to the open-source software we produced throughout our work and make them publicly available to the community.

4.1 Analysis of state of the art

- *Survey of quality attributes.* We conducted an extensive literature review of attributes describing quality in crowdsourcing. These attributes are split into 3 main dimensions: data , task and people related attributes [20].

- *Survey of assessment techniques.* We conducted an extensive literature review of techniques to assess quality in crowdsourcing. These techniques are grouped based on an actor doing the assessment: individual, group and computational [20].

- *Survey of assurance techniques.* We conducted an extensive literature review of techniques to assure high level of quality in crowdsourcing. These techniques are grouped according to strategies they utilize: improve data quality, select people, incentivize people, train people, improve task design and control execution [20].

- *Survey of approaches for crowdsourcing complex processes.* We created a framework and used it to evaluate 11 approaches to design and deploy complex crowdsourcing processes [76].

4.2 Empirical studies

- *Study about reward schemes and task design.* Conducting experiments on MTurk we identified that homogeneous reward schemes (focused on accuracy or speed) work better than heterogeneous (focused both on accuracy and speed). Asking workers to perform tasks faster, without introducing a financial motivation works. Tasks with simple user interfaces (lower cognitive demand) result in better accuracy of results [38].

- *Study about tasks searching experience.* From our experiments conducted on CrowdFlower, we identified that more than 40% of workers spend more than 25% of their time on the platform searching for tasks to work on. About 33% of workers spend 1-2 minutes to find a new task, 24% of workers spend more than 5 minutes. More experienced workers focus on tasks with more units to stay working longer [74].

- *Study about performance in mobile platforms.* From our experiments conducted on CrowdCafe, we identified that workers performing image labeling tasks are significantly less likely to produce random responses working on mobile devices, rather than on desktop. We believe that typing random words on smartphone takes the same time than appropriate ones, also on copy and paste is performed harder in mobile devices [72].

- *Study about execution patterns.* From our experiments conducted on CrowdFlower and MTurk, we identified that execution delays on CrowdFlower are primarily caused by abandoned assignments. In MTurk it workers join the task in several minutes since its publication, while in CrowdFlower it workers join in several seconds. Some workers on MTurk start several assignments and then work on them

sequentially. This kind of pattern is impossible on CrowdFlower. In MTurk fewer workers are involved doing several assignments each. In CrowdFlower a lot of workers are involved, therefore each performing only one or two assignments [75].

4.3 Interventions

- *Approaches to predict abandoned assignments* – simple linear regression can be applied to predict the duration of the last assignment. Using this as an assignment limit and relaunching during runtime all assignments active for longer times lead to several times improvement of overall task execution speed with only about 10% extra cost caused by false positives in predictions. Decision trees based on workers' in-page activity can help to do more accurate predictions of abandoned assignments [73, 77].

- *Approaches to predict low quality assignments.* It is possible to predict accuracy of outputs during runtime based on in-page activity of workers and a small ground truth data injected using machine learning algorithms (e.g. decision trees, random forests) [73].

4.4 Software prototypes

- *CrowdComputer*[10] [118] – is a crowdsourcing platform supporting execution of human, machine and crowd tasks. Complex crowdsourcing processes of these task types could be designed using our BPMN extension in Eclipse plugin and later deployed on CrowdComputer. We also provide a support for task streaming extending BPMN with a crowd task and a streaming connector.

[10] `https://github.com/crowdcomputer/`

- Task Listing Page[11] [74] – is a prototype of a task listing page designed to help workers spend less time to search tasks to work on. CrowdFlower implemented their new task listing page based on this prototype.

- *CrowdCafe*[12] [72] – is a mobile crowdsourcing platform where workers can perform tasks during semi-occupied situations (e.g. riding a bus, traveling by train, waiting in a queue) on smartphones in exchange for tangible rewards (e.g. a coffee, a bus ticket) provided in local stores.

- *ReLauncher*[13] [77, 73] – is a system for monitoring microtasks execution and relaunching assignments which are predicted as abandoned or of low quality.

5 Conclusions

As it is summarized in Section 4, we made contributions analyzing existing state of the art with several literature reviews [76, 20], conducting several empirical studies [38, 74, 117, 75], introducing approaches improving different quality dimensions [77, 73] and implementing several technical tools [118, 74, 72, 77, 73] to validate the approaches we introduced. These contributions were published in several peer-reviewed international conferences and journals, including ACM CSCW and AAAI HCOMP, which are the key conferences in the area of microtask crowdsourcing. We believe that our contributions leave a significant footprint in the domain of quality control in microtask crowdsourcing.

[11] Stays in a private repository of CrowdFlower
[12] `https://github.com/crowdcafe/`
[13] `https://github.com/relauncher/`

5.1 Limitations and future work

Our work has a set of limitations, which we consider as areas for future improvement. These limitations are discussed below.

- *Platforms.* We conducted our experiments on CrowdFlower, MTurk, CrowdCafe and CrowdComputer. Still there is a variety of other platforms available, such as ClickWorker and Microworkers. Different platforms support different motivations, attract different demographics of the crowd resulting in different workers behavior. From the technical perspective, platforms provide different freedom for requesters to design tasks, preselect workers, control task execution and reward workers. Therefore the results and conclusions we identified in our experiments might be not completely valid for other crowdsourcing platforms.

- *Task types.* We made conclusions in our experiments based on a limited set of task types (e.g. transcription, image labeling and business address search). Workers behave differently performing different task types, therefore the results we came up with could be not completely applicable for other task types.

- *Dataset size.* We experimented with tasks having 100 or fewer instances. We believe that execution patterns in very big tasks, having hundreds and thousands of data units, could be different, therefore the conclusions we make might be less valid for bigger datasets.

- *Predictors.* In our approaches to predict abandoned assignment and assignments having low accuracy we trained machine learning algorithms based on a fixed set of features we extracted from logging workers browser activity. We acknowledge that the set of features

used in the study may not be complete and therefore the best predictors are not yet in this set.

The limitations described above are common problems in the academic community focused on microtask crowdsourcing. We believe that there is a need for a general framework for mapping experiment results from one crowdsourcing platform to another in addition to ordinary cross platform experiments. There is also a need for a framework, mapping different task types with associated workers behavior (of a different quality level), used independently of platforms tasks are deployed on. The predictors derived from workers in-page activity should be adjusted according to workers' screen sizes, as they imply different behavior (e.g. more scrolling on mobile devices, easier copy and paste on desktop devices). It is also important to have such predictors, that even if workers are aware of the decision tree behind the automatic quality assessment, they should not be able to adjust their behavior to keep submitting random responses.

5.2 Vision for future

Different stakeholders in microtask crowdsourcing have different visions about where and how microtasking should evolve. We believe that for microtask crowdsourcing it is not yet time to work on fine tuning its different attributes (e.g. to identify the best time to post a task on a crowdsourcing platform or the most appropriate image size for image labeling tasks), but it is still time to do experiments with structural changes (e.g. to identify the most appropriate motivation strategy or the most convenient way for workers to execute tasks, such as a web-application, a native desktop application, a smart-TV application, an e-reader application or even screens on bus stops to get free tickets) in order to identify the most appropriate form microtask crowdsourcing can have. In this section we speculate

and discuss our vision of microtask crowdsourcing both from workers and requesters perspective.

Workers perspective

We do the following discussion making an assumptions, that microtask crowdsourcing is *not a full-time job*, therefore, there is no reason to judge it as a regular job and to compare wages with full-time jobs. Microtask crowdsourcing is more a way to improve a financial situation during free time, to learn new skills and to enjoy the process. If one is looking to work online for more than 3 hours a day with the primary goal to make money, it is better to look for freelance or part-time jobs than into microtask crowdsourcing. Below we discuss how workers experience could be improved by introducing changes in crowdsourcing platforms, task user interfaces, task assignment procedures and motivations.

Mobile platforms We believe that there is a big potential in mobile microtask crowdsourcing, so that people can perform tasks whenever and wherever they feel to do so in shifts of 5-60 minutes. Nowadays on the market there are various Android smartphones, primarily manufactured by China-based companies (e.g. Mi[14] and One Plus[15]), which are not expensive (e.g. some are below 250 USD) and are very powerful having big full-HD screens. We strongly believe that these kinds of smartphones have a great potential to become microtasking workplaces for people. Well designed mobile crowdsourcing platforms are needed. Currently there are not that many platforms available for online mobile work (e.g. Crowdee[16]), while there are plenty of them already available for offline work (e.g. BeMyEye[17]).

[14] http://www.mi.com/en/
[15] https://oneplus.net
[16] https://www.crowdee.de/en/
[17] https://uk.bemyeye.com

APIs for workers The existing crowdsourcing platforms, such as MTurk and CrowdFlower, have APIs for requesters to enrich their experience, still there is nothing like this for workers. If it is possible to perform all worker actions through APIs, then third-party developers could improve workers experience. Currently it is only possible to enrich workers experience by implementing browser-extensions, such as Turkopricon [57].

User interface If a requester thinks that a given microtask can not be performed on a smartphone, it is needed to rethink the user interface of the task or to admit that this is not a microtask and therefore should target instead freelance platforms. It is important to implement tasks not just for smaller screens, so that the screen size becomes a limitation, but to utilize all best-practices found in mobile application development, leveraging different screen transitions and input gestures (e.g. swipe for action, originally proposed by Mailbox[18]). It is interesting to conduct experiments with new ways of interaction for performing microtasks, such as voice interaction (to allow people with vision impairments to perform tasks), gestures (shaking, swiping, waving, rotating), messenger/chat (a personal assistant style), camera recording (to recognize worker gestures or movements to simplify the task execution) and others.

Assignment procedure We believe that tasks in crowdsourcing platforms could be treated as songs in music streaming services. When one song finishes, another one from the same album, artist or the similar style starts playing at Spotify[19] or Pandora[20]. Next tasks in crowdsourcing platforms could be streamed in a similar fashion, so workers stay performing and do not stay searching, which is beneficial for both requesters and workers. As

[18] `https://www.mailboxapp.com`
[19] `https://www.spotify.com`
[20] `http://www.pandora.com`

in Spotify in case of not-premium account music stream can be interrupted with ads, on crowdsourcing platforms task streams could be interrupted with high-priority tasks, potentially providing higher rewards. In addition developing a general extensible framework of obtainable skills is required not only to assign tasks to appropriate workers but also to motivate workers to learn.

Rewards Prefixed general reward strategies should be applied to lower the cognitive demand workers experience weighing tasks according to their financial benefit. Workers should know that they are treated well and compensated fairly according to the amount of plausible results they produce per unit of time. As we do not consider microtask crowdsourcing as a primary source of income, the rewards should be appropriate. Instead of paying cash (implying tax issues), it could be possible to compensate workers with bus pass credits, credits for utility expenses (e.g. water, electricity), smartphone credit, free subscriptions for online (e.g. Spotify, Netflix, iTunes, Udacity classes) or offline services (e.g. a medical insurance, school services).

Public profiles Worker profiles on crowdsourcing platforms could become as valuable as profiles in Stack Overflow[21], Behance[22] and Github[23] for future employment opportunities. There is a clear benefit of having complete profiles, rather than anonymous ones.

Human interactions We believe that workers in crowdsourcing platforms should be specifically treated as humans, rather than computational units. Inter workers collaboration could bring more emotional support and mo-

[21] http://stackoverflow.com
[22] https://www.behance.net
[23] https://github.com

tivation. Experienced workers could provide mentorship to newcomers, to guide them through their experience, to teach them best practices. Direct interactions between workers and requesters could bring more responsibilities on both sides.

Requesters perspective

In this subsection we address two important aspects for requesters: quality control and crowdsourcing complex work. We propose to address the quality control in two ways: analyzing workers behavior (an evolution of what we did in this thesis) and introducing higher responsibilities of workers and requesters on the platform.

Workers behavior analysis The quality control of results is not only a duty of requesters but crowdsourcing platforms in general. According to our experience workers are inconsistent in the quality of results they produce. We have an evidence that in some assignments selected workers produced very good results and in others provided random ones. Solid user behavior analysis techniques should be adopted in order to create clusters of workers behavior to predict if a given assignment is of a low quality, so further actions could be taken against it (e.g. relaunching).

Responsibility The problem of adversarial workers submitting random results could be partly solved by demotivating low quality work, which means not only the workers do not get a reward, but also their credits are subtracted with a small fine (in case an example of clear cheating is identified). Such workers accountability for quality of the results will make automatic bots generate negative value, and therefore they will not be introduced on such platforms. The same small fines could be applied to requesters submitting inconsistent tasks which workers feel not comfortable performing.

Complex crowdsourcing processes As we learned from our survey of approaches for designing and deploying complex crowdsourcing processes, appropriate tools exist already, but require knowledge of specific notations. To let a general audience of requesters to design and deploy their complex processes we might look out to other industries. Zapier[24] and IFTTT[25] provide decision-based rules to perform actions in and by independent integrated online services (e.g. when an image is posted to Instagram, save it to Dropbox). A tight integration with Zapier and IFTTT might help to design complex crowdsourcing processes in a user friendly way, where no knowledge of a professional or a proprietary notation is required. Such integrations could allow running crowdsourcing tasks without having an explicit crowdsourcing platform, as tasks could be performed in Google Spreadsheets, as comments in Dropbox, or image labels in Flickr.

5.3 Final remarks

Microtask crowdsourcing has a great potential to help millions of people around the world to gain new and master existing skills, to support their financial situation and to get more use out of their free time. It can help thousands of companies to have manual still scalable data collection and processing on demand. We encourage people from various communities including Human Computer Interaction, Computer Supported Cooperative Work, Software Engineering, Machine Learning, Financial Technologies and Mobile Application Development to unite their experience in building the next microtask crowdsourcing platform, potentially based on our vision discussed in Section 5.2.

[24] https://zapier.com
[25] https://ifttt.com

Bibliography

[1] B Thomas Adler and Luca De Alfaro. A content-driven reputation system for the wikipedia. In *WWW 2007*, pages 261–270, 2007.

[2] B Thomas Adler, Luca De Alfaro, Santiago M Mola-Velasco, Paolo Rosso, and Andrew G West. Wikipedia vandalism detection: Combining natural language, metadata, and reputation features. In *Computational linguistics and intelligent text processing*, pages 277–288. Springer, 2011.

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

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

[5] M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H.R. Motahari-Nezhad, E. Bertino, and S. Dustdar. Quality control in crowdsourcing systems: Issues and directions. *Internet Computing, IEEE*, 17(2):76–81, March 2013.

[6] M. Allahbakhsh, S. Samimi, H.-R. Motahari-Nezhad, and B. Benatallah. Harnessing implicit teamwork knowledge to improve quality in crowdsourcing processes. In *SOCA 2014*, pages 17–24, Nov 2014.

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

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

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

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

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

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

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

[14] Michael S. Bernstein, David R. Karger, Robert C. Miller, and Joel Brandt. Analytic methods for optimizing realtime crowdsourcing. *CoRR*, abs/1204.2995, 2012.

[15] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *UIST 2010*, pages 313–322. ACM, 2010.

[16] 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. Vizwiz: Nearly real-time answers to visual questions. In *UIST 2010*, UIST '10, pages 333–342, 2010.

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

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

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

[20] Cinzia Cappiello, Pavel Kucherbaev, Florian Daniel, Boualem Benatallah, and Mohammad Allahbahsh. Quality control in crowdsourcing: A survey of quality attributes, assessment techniques and assurance actions. *pre-print http://hdl.handle.net/11572/141045*, 2017.

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

[22] Lydia B. Chilton, John J. Horton, Robert C. Miller, and Shiri Azenkot. Task search in a human computation market. In *HCOMP 2010*, pages 1–9, 2010.

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

[24] R. Dawson and S. Bynghall. *Getting Results from Crowds: The Definitive Guide to Using Crowdsourcing.* Advanced Human Technologies, 2011.

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

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

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

[28] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G. Ipeirotis, and Philippe Cudr-Mauroux. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi, editors, *WWW (Companion Volume)*, page 617. ACM, 2015.

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

[30] Reddit discussion thread. In a five day turk-week, how much do you make? `https://www.reddit.com/r/mturk/comments/1hbnd6/in_a_five_day_turkweek_how_much_do_you_make/`. Accessed: 2016-02-02.

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

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

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

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

[35] Enrique Estellés-Arolas and Fernando González-Ladrón-De-Guevara. Towards an integrated crowdsourcing definition. *J. Inf. Sci.*, 38(2):189–200, April 2012.

[36] Boi Faltings, Radu Jurca, Pearl Pu, and Bao Duy Tran. Incentives to counter bias in human computation. In *Proceedings of the Seconf AAAI Conference on Human Computation and Crowdsourcing,*

HCOMP 2014, November 2-4, 2014, Pittsburgh, Pennsylvania, USA, 2014.

[37] Meng Fang, Jie Yin, and Dacheng Tao. Active learning for crowdsourcing using knowledge transfer. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[38] Ailbhe Finnerty, Pavel Kucherbaev, Stefano Tranquillini, and Gregorio Convertino. Keep it simple: Reward and task design in crowdsourcing. In *Proceedings of the Biannual Conference of the Italian Chapter of SIGCHI*, page 14. ACM, 2013.

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

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

[41] Trent Hamm. Tips for making money with amazon mechanical turk. `http://www.thesimpledollar.com/can-you-actually-earn-reasonable-money-from-mechanical-turk/`. Accessed: 2016-02-02.

[42] Derek L Hansen, Patrick J Schone, Douglas Corey, Matthew Reid, and Jake Gehring. Quality Control Mechanisms for Crowdsourcing: Peer Review, Arbitration, & Expertise at FamilySearch Indexing. In *CSCW 2013*, pages 649–660, 2013.

[43] Jeffrey Heer and Michael Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *CHI 2010*, pages 203–212, 2010.

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

[45] Paul Heymann and Hector Garcia-Molina. Turkalytics: analytics for human computation. In *WWW 2011*, pages 477–486, 2011.

[46] D. Hiemstra. A probabilistic justification for using tf.idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139, 2000.

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

[48] T. Hossfeld, C. Keimel, and C. Timmerer. Crowdsourcing quality-of-experience assessments. *Computer*, 47(9):98–102, Sept 2014.

[49] Jeff. Howe. The rise of crowdsourcing. *Wired*, June 2006.

[50] Chang Hu, Philip Resnik, Yakov Kronrod, and Benjamin Bederson. Deploying monotrans widgets in the wild. In *CHI 2012*, pages 2935–2938, 2012.

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

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

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

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

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

[56] Panos Ipeirotis. Why people participate on mechanical turk, now tabulated. `http://www.behind-the-enemy-lines.com/2008/09/why-people-participate-on-mechanical.html`. Accessed: 2016-02-02.

[57] Lilly C Irani and M Silberman. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *CHI 2013*, pages 611–620, 2013.

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

[59] Oliver P John, Laura P Naumann, and Christopher J Soto. Paradigm shift to the integrative big five trait taxonomy. *Handbook of personality: Theory and research*, 3:114–158, 2008.

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

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

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

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

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

[65] Shashank Khanna, Aishwarya Ratan, James Davis, and William Thies. Evaluating and improving the usability of mechanical turk for low-income workers in india. In *Proceedings of the first ACM symposium on computing for development*, page 12. ACM, 2010.

[66] Roman Khazankin, Daniel Schall, and Schahram Dustdar. Predicting qos in scheduled crowdsourcing. In *CAISE 2012*, pages 460–472, 2012.

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

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

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

[70] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *CSCW 2013*, pages 1301–1318, 2013.

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

[72] Pavel Kucherbaev, Azad Abad, Stefano Tranquillini, Florian Daniel, Maurizio Marchese, and Fabio Casati. CrowdCafe: mobile crowdsourcing platform. Technical report, University of Trento, Department of Information Engineering and Computer Science, 11 2014.

[73] Pavel Kucherbaev, Florian Daniel, and Maurizio Marchese. Relaunching strategies to improve speed and accuracy of crowdsourcing microtasks. *pre-print http://hdl.handle.net/11572/141025*, 2017.

[74] Pavel Kucherbaev, Florian Daniel, Maurizio Marchese, Fabio Casati, and Brian Reavey. Toward effective tasks navigation in crowdsourc-

ing. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, pages 401–404. ACM, 2014.

[75] Pavel Kucherbaev, Florian Daniel, Stefano Tranquillini, and Maurizio Marchese. Modeling and exploration of crowdsourcing micro-tasks execution. In *Third AAAI Conference on Human Computation and Crowdsourcing*, 2015.

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

[77] Pavel Kucherbaev, Florian Daniel, Stefano Tranquillini, and Maurizio Marchese. ReLauncher: Crowdsourcing Micro-Tasks Runtime Controller. In *CSCW 2016*. To appear, 2016.

[78] Pavel Kucherbaev, Stefano Tranquillini, Florian Daniel, Fabio Casati, Maurizio Marchese, Marco Brambilla, and Piero Fraternali. Business processes for the crowd computer. In *Business process management workshops*, pages 256–267. Springer Berlin Heidelberg, 2012.

[79] A. Kulkarni, P. Gutheim, P. Narula, D. Rolnitzky, T. Parikh, and B. Hartmann. Mobileworks: Designing for quality in a managed crowdsourcing architecture. *Internet Computing, IEEE*, 16(5):28–35, Sept 2012.

[80] Anand Kulkarni, Matthew Can, and Björn Hartmann. Collaboratively crowdsourcing workflows with turkomatic. In *CSCW 2012*, pages 1003–1012, 2012.

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

[82] Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. Real-time captioning by groups of non-experts. In *UIST 2012*, pages 23–34. ACM, 2012.

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

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

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

[86] John Le, Andy Edmonds, Vaughn Hester, and Lukas Biewald. Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution. In *SIGIR 2010 workshop on crowdsourcing for search evaluation*, pages 21–26, 2010.

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

[88] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Exploring iterative and parallel human computation processes. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 68–76. ACM, 2010.

[89] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Turkit: human computation algorithms on mechanical turk. In *UIST 2010*, pages 57–66. ACM, 2010.

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

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

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

[93] Tim Matthews. State of enterprise crowdsourcing. `https://www.crowdflower.com/blog/2013/11/the-state-of-enterprise-crowdsourcing`. Accessed: 2016-02-02.

[94] R.R. Morris, M. Dontcheva, and E.M. Gerber. Priming for better performance in microtask crowdsourcing environments. *Internet Computing, IEEE*, 16(5):13–19, Sept 2012.

[95] Jakob Nielsen, Marie Tahir, and Marie Tahir. *Homepage usability: 50 websites deconstructed*, volume 50. New Riders Indianapolis, IN, 2002.

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

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

[98] C. Puah, A.Z.A. Bakar, and Chu Wei Ching. Strategies for community based crowdsourcing. In *ICRIIS 2011*, pages 1–4, Nov 2011.

[99] Alexander J Quinn and Benjamin B Bederson. Asksheet: Efficient human computation for decision making with spreadsheets. In *CSCW 2014*, pages 1456–1466, 2014.

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

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

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

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

[104] Jeffrey Rzeszotarski and Aniket Kittur. Crowdscape: interactively visualizing user behavior and output. In *UIST 2012*, pages 55–62. ACM, 2012.

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

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

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

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

[109] Daniel Schall, Benjamin Satzger, and Harald Psaier. Crowdsourcing tasks to social networks in bpel4people. *World Wide Web*, 17(1):1–32, 2014.

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

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

[112] Yaron Singer and Manas Mittal. Pricing mechanisms for crowdsourcing markets. In *WWW 2013*, pages 1157–1166, 2013.

[113] Yu-An Sun and Christopher Dance. When majority voting fails: Comparing quality assurance methods for noisy human computation environment. *arXiv preprint arXiv:1204.3516*, 2012.

[114] James Surowiecki. *The Wisdom of Crowds*. Anchor, 2005.

[115] O. Tokarchuk, R. Cuel, and M. Zamarian. Analyzing crowd labor and designing incentives for humans in the loop. *Internet Computing, IEEE*, 16(5):45–51, Sept 2012.

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

[117] Stefano Tranquillini, Florian Daniel, Pavel Kucherbaev, and Fabio Casati. Bpmn task instance streaming for efficient micro-task crowdsourcing processes. In *Business Process Management*, pages 333–349. Springer International Publishing, 2015.

[118] Stefano Tranquillini, Florian Daniel, Pavel Kucherbaev, and Fabio Casati. Modeling, enacting, and integrating custom crowdsourcing processes. *ACM Transactions on the Web (TWEB)*, 9(2):7, 2015.

[119] Donna Vakharia and Matthew Lease. Beyond amt: An analysis of crowd work platforms. *CoRR*, abs/1310.1672, 2013.

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

[121] Fernanda B Viégas, Martin Wattenberg, and Matthew M McKeon. The hidden order of wikipedia. In *Online communities and social computing*, pages 445–454. Springer, 2007.

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

[123] Maja Vukovic and Claudio Bartolini. Towards a research agenda for enterprise crowdsourcing. In *Leveraging Applications of Formal Methods, Verification, and Validation*, volume 6415 of *LNCS*, pages 425–434. Springer, 2010.

[124] Bo Waggoner and Yiling Chen. Output agreement mechanisms and common knowledge. In *HCOMP 2014*, 2014.

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

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

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

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

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

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

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

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

Appendix A

CrowdCafe: Mobile Crowdsourcing Platform

technical report

CrowdCafe - Mobile Crowdsourcing Platform

Pavel Kucherbaev
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
pavel.kucherbaev@unitn.it

Azad Abad
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
azad.abad@unitn.it

Stefano Tranquillini
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
stefano.tranquillini@unitn.it

Florian Daniel
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
florian.daniel@unitn.it

Maurizio Marchese
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
maurizio.marchese@unitn.it

Fabio Casati
University of Trento
Via Sommarive 9, Povo (TN)
38123, Italy
fabio.casati@unitn.it

ABSTRACT
In this paper we present a mobile crowdsourcing platform CrowdCafe, where people can perform microtasks using their smartphones while they ride a bus, travel by train, stand in a queue or wait for an appointment. These microtasks are executed in exchange for rewards provided by local stores, such as coffee, desserts and bus tickets. We present the concept, the implementation and the evaluation by conducting a study with 52 participants, having 1108 tasks completed.

ACM Classification Keywords
H.5.3 Group and Organization Interfaces: Computer-supported cooperative work

General Terms
Human Factors, Design, Performance

Author Keywords
Crowdsourcing, User Interfaces, Motivation

INTRODUCTION

Crowdsourcing is the practice of outsourcing work to an unknown group of people via the Internet, instead of assigning it to internal employees [7]. Crowdsourcing has been so far very successful in performing tasks which are still hard to automate using algorithms, while they can be relatively easily solved by humans, such as image object recognition, annotations, feedback collection and similar.

Requestors are the people who want to crowdsource their work. They publish tasks on crowdsourcing platforms where requestors meet potential workers - people who solve tasks for monetary reward, curiosity or other motivations. Some examples of crowdsourcing platforms are Amazon Mechanical Turk (MTurk), CrowdCloud, MicroWorkers, Mobileworks, CrowdFlower. In general workers need to perform tasks from their desktops or laptops, as tasks are designed for non-mobile screens. However, many people, when they have time with their computer, prefer to perform some things related to their job or just to have fun watching something.

People spend everyday some amount of time riding a bus, standing in a line at a grocery store, waiting for a doctor appointment. During this time they can read a pocket book or use their smartphones. Many people end up checking their social network profiles. Mea et al. [12] showed an evidence that users can perform some tasks via mobile devices faster than via desktop.

We present a crowdsourcing platform CrowdCafe, where people perform microtasks, specifically designed for mobile execution, in exchange for non-monetary rewards provided in local stores, such as coffee, desserts and bus tickets.

STATE OF THE ART

The current research in the field of mobile crowdsourcing can be separated by three objectives: i) to help people from developing countries to earn extra cash, ii) to utilize smartphone sensors to collect location specific data and iii) to discover new concepts of performing crowdsourcing tasks.

Helping people from developing countries

Eagle et al. [3] and Kulkarni et al. [10] presented platforms (*txteagle* and *MobileWorks*), using which people from developing countries can earn extra money by completing various tasks using their mobile low-cost phones. Gupta et al. [5] presented a platform *mClerk* for mobile paid crowdsourcing in developing regions, which processes (sends and receives) tasks via SMS.

Mobile-sensing

Yan et al. [17] proposed an iPhone-based mobile crowdsourcing platform *mCrowd*, using which mobile users can perform tasks, using their smartphone sensors. Tamilin et al. [14] presented a context-aware crowdsourcing system for conducting crowdsourcing campaigns with smart phone users, which utilizes sensors available on mobile devices.

Discovering new concepts

Vaish et al. [16] presented an Android application *Twitch*[1], which in order to unlock a phone, asks its owner to answer a simple question, such as: how many people are around, or which activity the owner is doing now. Similarly, Truong et al. [15] showed how different crowdsourcing tasks can be completed using different unlocking gestures. Heimerl et al. [6] presented *Umati* – communitysourcing vending machine, which helps to attract a specific local group of people (e.g. people with deep knowledge in computer science) to perform tasks on the screen of the vending machine in exchange for snacks. Luon et al. [11] proposed a mobile system *Rankr* for crowdsourcing opinions via pair comparison of images and sentences on mobile phones. In [8] Kittur et al. analyzed how different aspects of crowdsourcing could be improved, they also challenged the community to revolutionize the conception of what a crowdsourcing platform is.

Musthag et al. [13] did an analysis of differences between mobile crowdsourcing platforms and desktop ones. They found a significant difference in demographics. The comparison was not straight in sense that mobile crowdsourcing platforms mostly support offline location-dependent tasks, while desktop support online mainly. Mea et al. [12] conducted user studies where they tried to identify which crowdsourcing tasks suit better for mobile and which for desktop devices.

With CrowdCafe we aim:

- to investigate how during semi-occupied situations (such as riding a bus, traveling by train or waiting in a line) people can perform microtasks using their smartphones, not for the purpose of making income, but to have fun and to benefit out of this time,
- to boost the research in the mobile crowdsourcing field, by providing to the academic community an open-sourced platform which is deployed online, so other researchers can conduct studies and extend the platform if needed,
- to identify the best practices of designing tasks user interfaces and to create a repository of reusable user interface patterns.

CONCEPT

The concept of CrowdCafe affects three aspects of crowdsourcing: *tasks user interfaces*, *tasks classification* and *workers motivation*.

Tasks User Interfaces

In order to provide a good user experience of tasks execution on CrowdCafe we want to apply the best user interface (UI) practices from current mobile applications, such as: *feed* to present all the content as a list, without sidebars; *big full width buttons* to make it comfortable to press them with a thumb; *swipe for action* to keep a user interface very clean without buttons, where, depending on a direction and a distance of swiping a UI element, different actions are triggered (was announced with the MailBox mobile application[2]).

[1] `http://twitch.stanford.edu/`

[2] `http://www.mailboxapp.com/`

Tasks Classification

As described by [9] on the platforms such as MTurk or CrowdFlower it is hard for workers to select a task to work on, because descriptions are not informative enough and they never know how much time they will spend on execution. On CrowdCafe we decided to split all the tasks by completion time in 3 clear categories:

- *"Espresso"* - about 10 seconds to be completed, with mostly only clicking and swiping actions required (e.g. to identify the sentiment of tweets or to compare pairs of images);
- *"Cappuccino"* - about 2 minutes to be completed, with some typing and learning required (e.g. to fill up a short survey, to annotate images);
- *"Wine"* - more than 5 minutes to be completed, might require a worker to be in a specific context or a location (e.g. to go to a grocery store and to make a photo of a particular product with its price).

Workers Motivation

We do not position CrowdCafe as a source for primary or secondary income. We consider CrowdCafe as a way to convert time, which is wasted otherwise, to enjoyable rewards. According to Dan Ariely [1] the smaller reward now is more desirable than a bigger one later. So we want to minimize the time frame between a worker starting task execution and a worker enjoying a reward, by providing micro rewards from local stores, such as coffee or dessert at a university bar. This can help workers to start working on tasks and in 15 minutes to feel an outcome of their work by drinking a coffee.

IMPLEMENTATION

We have implemented the CrowdCafe as a website[3]. It has two main components: i) "Kitchen", where requestors design and publish tasks from a desktop; ii) "Cafe", where workers perform tasks using their mobile phones and get reward coupons. The CrowdCafe code is open-sourced and is available on GitHub[4], where more details about the implementation can be found.

Requestors Interface

In "Kitchen" requestors create tasks (Figure 1), defining: 1) title, 2) instructions, 3) category, 4) preselection logic, 5) user interface, 6) input dataset and 7) quality control settings. The first three parts are trivial and we focus on the other four.

Preselection

Requestors can define to which workers the task will be visible by adding a set of restrictions, such as "worked" or "did not work" on particular tasks. This simple preselection logic is powerful enough to route surveys to particular workers, to create tasks which are only visible to workers who performed some skill test tasks.

[3] `http://crowdcafe.io/`

[4] `https://github.com/CrowdCafe/crowdcafe`

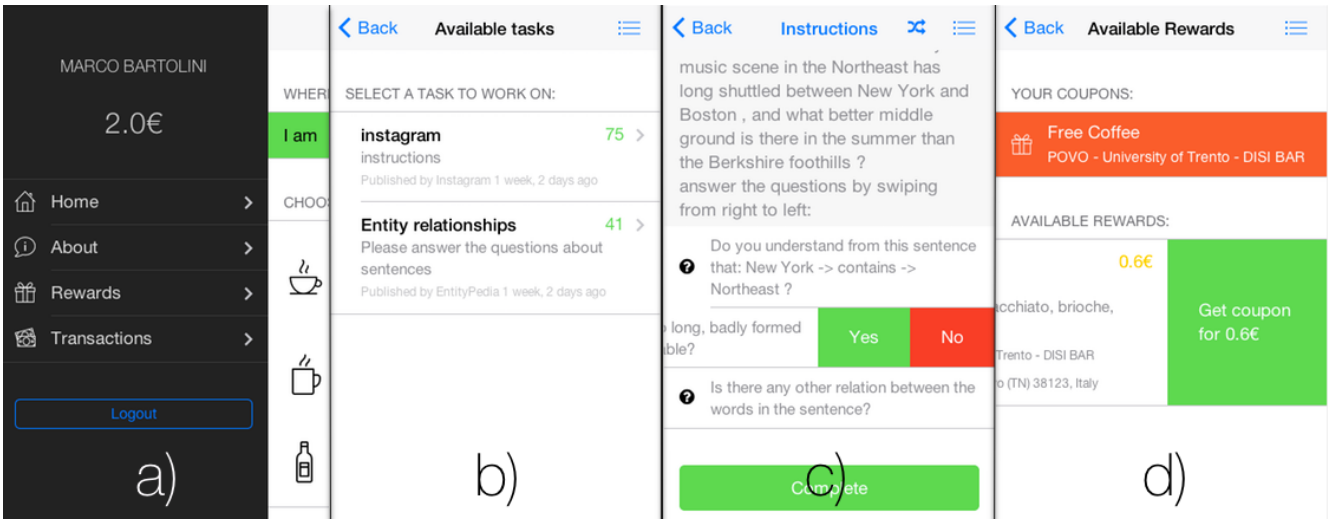

Figure 2. Workers User Interface. a) Task categories and side menu, b) Available tasks, c) Example of a task, d) Reward page

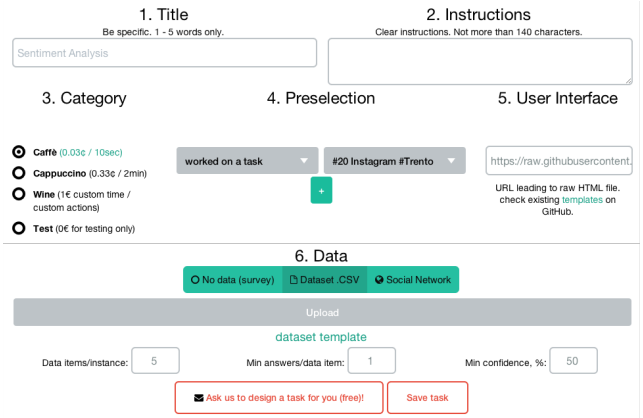

Figure 1. Requestors user interface

User Interface

We decided to leave a lot of freedom for requestors, so they can design the UI of their tasks using HTML and CSS. In order to not design from scratch and to accumulate the best practices such UI templates are stored in the public GIT repository. To apply a particular template, requestors need to insert its URL, which refers to a raw HTML file.

Input Dataset

There are three options for input data: 1) no input data (a survey task), 2) data uploaded from a .csv file, 3) data from a social network feed (e.g. Twitter, Instagram) on a particular topic, defined by a hash-tag (such as #helsinki). Requestors also define how many data units (rows in case of .csv file, or tweets in case of Twitter) a worker should process in one time.

Quality Control

Requestors can define a similarity function to check whether a given judgement is similar to gold data (predefined correct answer) or whether judgements given by different workers are similar to each other (an agreement is found). By default this similarity function is a simple equality, while requestor can create a script which does more complex similarity assessment.

Gold units (if available) are injected into tasks with a probability calculated by the formula:

$$p = \frac{1 + N_{incorrect}}{1 + N_{incorrect} + N_{correct}} \quad (1)$$

, where $N_{incorrect}$ – number of incorrect judgements for gold units given by a worker, $N_{correct}$ – number of correct judgements for gold units given by a worker. The more correct judgements a worker gives, the less probability of having them further injected he has.

Requestor can also define a limit of mistakes a worker is allowed to make (which is zero by default).

Workers Interface

The mobile website for tasks execution has four main sections: 1) home page and side menu (Figure 2a), 2) tasks listing page (Figure 2b), 3) task execution page (Figure 2c), 4) reward page (Figure 2d).

Home page

On the home page workers see the list of task categories. In the right part of the top bar on every page there is a button which opens the side menu. There is a context select box where we ask workers to define where they currently are (e.g. in a bus, having lunch). On the side menu workers see their name, the amount of money they have earned and four links: 1) “Home”, 2) “About”, where the service is described in details, 3) “Rewards”, 4) “Transactions” - where the history of all earnings (related to tasks execution) and spendings (related to purchasing coupons) is presented. This side menu also has a logout button.

Tasks listing page

On the tasks listing page workers see a list of tasks with its description and amount of instances available.

Task execution page
When workers start executing a task, they first see a pop up window with instructions. Workers are expected to read them and after they can start to perform a task. Workers should fill up all the necessary fields otherwise they can not submit the task. After they submit one task, they are redirected to another instance of this task. If there are no any instances available workers are redirected back to the tasks listing page.

Rewards page
On this page a worker can see a list of available rewards with its price and address where they can get them.

EVALUATION

In order to evaluate the concept of CrowdCafe along with our implementation we decided to post two tasks and one follow up survey:

- "Instagram #Trento" – in this task workers were expected to look on images potentially about Trento and provide two actions: 1) add three relevant tags to each image, 2) specify whether this image really represents Trento as a city.

- "Sentence Analysis" – in this task workers were expected to read a short text and to answer two questions: 1) is a given relationship between two nouns correct from the content of the text? (yes, no, i don't know) 2) does the text consist of only one clear sentence? (yes, no)

- Survey – in this task we wanted to collect the workers feedback about tasks and the CrowdCafe platform in general.

For the first task we uploaded a dataset of 1000 sentences, splitting them in 334 tasks of 3 sentences, asking for at least 3 judgments for each sentence. For the second task we took a feed of 231 images from Instagram with a hashtag #trento, splitting them in 77 tasks of 3 images, asking for at least 3 judgments. Both tasks were qualified as "Espresso" tasks with a reward of 0.03 euro. The final survey was qualified as "Cappuccino" task with a reward of 0.33 euro.

We prepaid 84 coffees (0.60 euro each) at the bar on our faculty and left there a list of 84 unique codes. When workers earn enough money, they can purchase a coupon which they exchange for a coffee or a dessert at the bar.

In order to approach the first users we sent email invitations to people from our research group (30 people) and distributed 20 printed posters around our faculty building. It helped to get 80 sign ups on the platform in 1 day.

Results

We collected all the judgments for all tasks in two days from 52 workers. Two workers were identified which used the vulnerability in the code and submitted extra 400 equal judgments. These judgments were removed from the analysis. Some people did not specify a place where they performed tasks, so about a half (46.9%) of all responses did not have associated place. Out of those, which had: 56.40% were performed on a workplace, 14.10% – outside, 13.13% – in a bus, 11.83% – at home, 4.38% – in a train, 0.16% – walking.

Task 1 - "Instagram #Trento"
The average execution time for this task was 107.31 seconds (317 task responses, median 87, standard deviation 88.03 sec). We received 791 judgments for 231 image. There were 737 (93.17%) images according to instructions and included three or more tags, while 54 included only 1 or 2 tags.

Kappa evaluation metric shown to be accepted option to analyze the reliability of the inter agreement among workers[2]. We used Fleiss Kappa [4] to assess the reliability of the provided tags. We could not use the tags directly to estimate the Kappa values due to the open vocabulary of the tags provided by workers. Therefore, 3 experts (members of our research group) categorized all the tags into 10 predefined clusters and voting system has been used to select the ground truth cluster for each tag. Finally, cluster names were replaced with the real tags and Kappa values were calculated. The overall Fleiss Kappa value is 0.4416 with 0.0154 error that is an indication of a moderate agreement. The 95% confidence interval of Fleiss Kappa is [43.4, 44.9]. Also, the *p-value* is less than 0.0001 which shows that the observed agreement is statistically significant.

Task 2 - "Sentence Analysis"
The average execution time for this task was 62.85 seconds (1006 task responses, median 16 sec, standard deviation 276.76 sec). For each sentence we received from 3 to 5 judgments. These judgments have very low agreement level (we did not find any agreement in the majority of sentences).

Survey task
We sent email invitations to 18 people who provided at least one judgment to both "Instagram #Trento" and "Sentence Analysis" tasks. There are 15 people completed this survey. Out of these people 66% responded that 0.03 euro is enough reward for such tasks, 80% responded that they will use the platform in future, the average interest on scale from -3 (very negative) to +3 (very positive) in "Instagram #Trento" is 0.93, in "Sentence Analysis" is -0.60. The average overall satisfaction about CrowdCafe platform is 1.93 on the same scale.

Discussion

When we designed our two tasks we did not expect that it would take people so much time (107.31 and 62.85 seconds) to complete them. It showed that we classified these tasks not correct. Still we had workers, providing many judgments and in the survey the majority of workers responded, that 0.03 euro is enough reward for completing such tasks (which is only about 2 euro per hour). Several people mentioned that they performed tasks thinking about coffee and not money.

The overall quality of tags provided in "Instagram #Trento" task is high. All the tags were relevant. Even when workers did not follow the instructions and provided less than 3 tags or tags in other language than English (54 judgments), tags were still relevant. We found very low agreement between workers stating that an image characterizes Trento. There are two

possible reasons: 1) instructions were not clear enough and some people marked only images which have some famous Trento building on it, while others marked all images which they believed were made in Trento, 2) some users pointed an issue that this button did not work well in the native Android browser.

In the "Sentence Analysis" task there is very low agreement between workers, because many workers did not understand the instructions clearly and some workers did not pay enough attention and simply provided random results. This is also the reason of the big standard deviation and big difference between mean and median execution time for this task. In addition the survey results show that the interest in this task was much lower than in the "Instagram #Trento" task.

CONCLUSION

In this paper we have described the concept and evaluated the implementation of the mobile crowdsourcing platform CrowdCafe, where people can and are willing to perform microtasks during short spans of free time in exchange for tangible rewards such as coffee. We showed that for well-designed tasks with clear instructions, even without any specific control (e.g. gold data, skill tests), workers provide results of a very high quality (93.17%). In tasks with ambiguous instructions the quality of results is poor.

In future we plan to investigate: i) the variety of tasks people can perform on their smartphones with better or the same quality as on regular "desktop" crowdsourcing platforms (as an extension of Mea et al. [12] work), ii) the workers productivity with different user interface approaches in tasks design (e.g. radio buttons, swiping to action, set of buttons), iii) how different motivation strategies (e.g. cash, coffee, donation to charity, no reward) affect the workers productivity.

REFERENCES

1. Ariely, D., and Wertenbroch, K. Procrastination, deadlines, and performance: Self-control by precommitment. *Psychological Science 13*, 3 (2002), 219–224.
2. Carletta, J. Assessing agreement on classification tasks: the kappa statistic. *Computational linguistics 22*, 2 (1996), 249–254.
3. Eagle, N. txteagle: Mobile crowdsourcing. vol. 5623 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, 447–456.
4. Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychological bulletin 76*, 5 (1971), 378.
5. Gupta, A., Thies, W., Cutrell, E., and Balakrishnan, R. mclerk: Enabling mobile crowdsourcing in developing regions. CHI '12, ACM (New York, NY, USA, 2012), 1843–1852.
6. Heimerl, K., Gawalt, B., Chen, K., Parikh, T., and Hartmann, B. Communitysourcing: Engaging local crowds to perform expert work via physical kiosks. CHI '12, ACM (New York, NY, USA, 2012), 1539–1548.
7. Howe, J. The rise of crowdsourcing. *Wired 14*, 14 (October 2006), 1–7.
8. Kittur, A., Nickerson, J. V., Bernstein, M., Gerber, E., Shaw, A., Zimmerman, J., Lease, M., and Horton, J. The future of crowd work. CSCW '13, ACM (New York, NY, USA, 2013), 1301–1318.
9. Kucherbaev, P., Daniel, F., Marchese, M., Casati, F., and Reavey, B. Toward effective tasks navigation in crowdsourcing. AVI '14, ACM (New York, NY, USA, 2014), 1843–1852.
10. Kulkarni, A., Gutheim, P., Narula, P., Rolnitzky, D., Parikh, T., and Hartmann, B. Mobileworks: Designing for quality in a managed crowdsourcing architecture. *Internet Computing, IEEE 16*, 5 (Sept 2012), 28–35.
11. Luon, Y., Aperjis, C., and Huberman, B. Rankr: A mobile system for crowdsourcing opinions. vol. 95. Springer Berlin Heidelberg, 2012, 20–31.
12. Mea, V. D., Maddalena, E., and Mizzaro, S. Crowdsourcing to mobile users: A study of the role of platforms and tasks. In *DBCrowd*, R. Cheng, A. D. Sarma, S. Maniu, and P. Senellart, Eds., vol. 1025 of *CEUR Workshop Proceedings*, CEUR-WS.org (2013), 14–19.
13. Musthag, M., and Ganesan, D. Labor dynamics in a mobile micro-task market. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, ACM (New York, NY, USA, 2013), 641–650.
14. Tamilin, A., Carreras, I., Ssebaggala, E., Opira, A., and Conci, N. Context-aware mobile crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, ACM (New York, NY, USA, 2012), 717–720.
15. Truong, K. N., Shihipar, T., and Wigdor, D. J. Slide to x: Unlocking the potential of smartphone unlocking. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, ACM (New York, NY, USA, 2014), 3635–3644.
16. Vaish, R., Wyngarden, K., Chen, J., Cheung, B., and Bernstein, M. Twitch crowdsourcing: Crowd contributions in short bursts of time.
17. Yan, T., Marzilli, M., Holmes, R., Ganesan, D., and Corner, M. mcrowd: A platform for mobile crowdsourcing. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, ACM (New York, NY, USA, 2009), 347–348.