

Fair-Trade Crowdsourcing: Predicting the Working Times of Microtasks

公平取引クラウドソーシング：マイクロタスクの作業時間推定

February 2020

Susumu SAITO

齋藤 奨

Fair-Trade Crowdsourcing: Predicting the Working Times of Microtasks

公平取引クラウドソーシング：マイクロタスクの作業時間推定

February 2020

Research on Perceptual Computing
Department of Computer Science and Communications Engineering
Graduate School of Fundamental Science and Engineering
Waseda University

Susumu SAITO

齋藤 奨

Fair-Trade Crowdsourcing: Predicting the Working Times of Microtasks

by

Susumu SAITO

Submitted to the Department of Computer Science and Communications Engineering
in February 2020, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This dissertation presents a series of research work regarding predicting working times of crowdsourcing microtasks, which are aimed towards assisting crowd workers estimate the lucrativeness of microtasks. In current crowd-working platforms, crowd workers are often underpaid. One of the main reasons is that it is difficult for certain workers to find microtasks that would pay well, by estimating “working time” — how long it would take to finish certain microtasks — based on various task-relevant data provided before starting them, and comparing it with price. This dissertation is comprised of three different works: *i*) investigation of worker strategy through tool and online community usage, *ii*) system architecture design for working time prediction, and *iii*) objective/evaluation functions design based on the perception to prediction errors of workers. I will start by describing a survey among crowd workers that inquire about their usual working strategies through questioning their knowledge and usage of third-party worker tools and online communities, followed by analyses of what types of worker assistance is currently appreciated and will be further needed, in order to emphasize the importance of working time prediction addressed in this study. I will then present a machine learning-based approach for predicting working times of crowd work microtasks. This study proposes solutions to several challenges in building a system, including the development of a browser tool for data collection in cooperation with workers, the definitions of four different recording methods of microtask working times employed in the browser tool, and a feature engineering problem for taking microtask-, worker-, and

requester-relevant information into account for input features. Finally, I will present an empirical methodology for quantifying the perception of workers to errors in working time prediction as well as for employing the perception measurement in building objective and evaluation functions used in a model for working time prediction. Employing such worker perception is expected to optimize and evaluate the predictive model based on how meaningful predicted outputs would be for workers' microtask selection. In order to achieve the foregoing, I conducted a survey among AMT workers to collect their impressions to presented prediction errors and to use the samples for formulating a relationship between working time and the maximum value of prediction error that workers would be able to tolerate. By using the derived relationship, my experimental results showed that my model was able to predict working times of $\sim 73\%$ of all the tested microtasks within workers' level of tolerance.

Defense Committee

Tetsunori Kobayashi	Professor, Faculty of Science and Engineering, Waseda University
Tetsuji Ogawa	Professor, Faculty of Science and Engineering, Waseda University
Tatsuo Nakajima	Professor, Faculty of Science and Engineering, Waseda University
Yukino Baba	Associate Professor, Faculty of Engineering, Information and Systems, Tsukuba University

Qualification Exam 3 Committee (for Graduate Program of Embodiment Informatics)

Tetsunori Kobayashi	Professor, Faculty of Science and Engineering, Waseda University
Tatsuo Nakajima	Professor, Faculty of Science and Engineering, Waseda University
Hiroyasu Iwata	Professor, Faculty of Science and Engineering, Waseda University
Shinji Asakura	Energy Products Country Manager, Tesla Motors Japan
Jeffrey P. Bigham	Associate Professor, School of Computer Science, Carnegie Mellon University

Acknowledgments

First of all, I would like to thank all of the thesis committee members: Prof. Tetsunori Kobayashi, Prof. Tatsuo Nakajima, Prof. Tetsuji Ogawa, and Dr. Yukino Baba of the defense committee; and Prof. Tetsunori Kobayashi, Prof. Tatsuo Nakajima, Prof. Hiroyasu Iwata, Mr. Shinji Asakura, and Dr. Jeffrey Bigham of the qualification exam 3 committee.

My deepest gratitude goes to Prof. Tetsunori Kobayashi, who had supported me as my supervisor for seven years since I began my first research in Perceptual Computing Lab as a third-year university student in 2014. He was always willing and enthusiastic in assisting me in any research activity I engaged in and in understanding my motivation behind it. He was also the first person to encourage me to join the graduate program for embodiment informatics, the opportunity that I decided to pursue a Ph.D. degree in. Without him, I could never have had such an adventurous and exciting academic life.

Next, I would like to express my sincere gratitude to Dr. Teppei Nakano for being my greatest mentor in our research team. Since the time I joined his team as a first-year graduate student, he had given me a plethora of insights to my research work through discussions, supported me in writing papers, and listened to my career-related concerns. I would also like to thank Prof. Tetsuji Ogawa, Mr. Makoto Akabane, and all of the students in the IoT research team who continuously followed up on my research progress and gave me helpful feedback.

My Ph.D. life has been very special thanks to the Graduate Program of Embodiment Informatics. I would like to thank the program coordinator, Prof. Shigeki Sugano, and all other program organizers for providing me with a lot of chances. The times I spent in Kobo had brought me a lot of conversations, collaborations, and opportunities to participate in off-campus activities and workshops. I am also very grateful for the financial support that enabled me to study abroad in UC Davis and in Carnegie Mellon University, making my Ph.D. life more fruitful. I would also like to thank all my colleagues in Kobo. Since they were nearly all of the Ph.D. students I knew in the campus (although they majored in many different disciplines), it had been a precious opportunity for me to have been able to share my thoughts and Ph.D.-related concerns with them.

My work could never have been done without my half-year internship in Carnegie Mellon University during my first year of Ph.D. studies. I would like to deeply thank Dr. Jeffrey Bigham,

my research advisor in CMU, since he has given me opportunities to broaden my horizon. All the things I learned and projects I was involved in at BigLab had greatly contributed to the progress of my research. Jeff was also kind enough to invite me to Skype meetings for discussions even after the internship, to introduce many other great crowd researchers, and to visit our lab back in Japan to catch up. I would also like to thank Toni Kaplan, Dr. Kotaro Hara, Chun-Wei Chiang, and Dr. Saiph Savage, the great co-authors of my papers who had a lot of contributions in the system development and/or in fruitful discussions with me, either remotely or in person. Also, I am grateful to all of the friends I made in CMU who made my stay in Pittsburgh unforgettable.

I would like to thank all the funds and fellowships that supported my research and living through my masters and Ph.D. I had been granted a fellowship by the Graduate Program of Embodiment Informatics from April 2015 to March 2019, and by JSPS Research Fellowship for Young Scientists (DC2) from April 2019 to February 2020. My research work was also partially supported by CASIO Science Promotion Foundation.

Finally, I would like to show my appreciation for all those who mentally supported me in private. My fiancée, Natsumi Ikeda, had always been very supportive at every moment; she listened to me talk about every joyful and depressing moment and gave me opportunities to take breathers from busy and tough days. And last but not least, I cannot thank my parents enough. With their continuous mental and financial support, I am quite sure that I am the best person I could be for now — and I am very proud to be the first Ph.D. degree holder in my family (as far as I am aware of).

Contents

1	Introduction	17
1.1	Context of Studies	17
1.1.1	Definition and Use of Crowdsourcing	17
1.1.2	Importance of Microtask Working Time Prediction	19
1.2	Background of Crowd Labor and Worker Assistance	22
1.2.1	Basic Worker Procedure	22
1.2.2	Unfair Payment in Crowd Markets	24
1.2.3	Available Online Communities and Worker Tools	27
1.2.4	Approach	29
1.3	Research Objectives	30
1.4	Dissertation Organization	33
2	Investigating Worker Strategies and Tool Use Among Crowd Workers	35
2.1	Introduction	36
2.2	Worker Survey Design	37
2.2.1	Survey Procedure	37
2.2.2	Survey Questions	37
2.2.3	Data Cleaning	40
2.3	Survey Results	40
2.3.1	Demographics	41
2.3.2	External Resource Usage	43

2.3.3	Time and Frustration	45
2.3.4	HIT Type Preference	47
2.3.5	Workers' Decision Making Criteria	48
2.4	Discussion	50
2.4.1	Available Assistancess Appreciated By Workers	51
2.4.2	Further Direction: Working Time Prediction	51
2.5	Conclusion	53
3	TurkScanner: Microtask Hourly Wage Prediction	55
3.1	Introduction	55
3.2	Related Work	57
3.3	Measuring Working Time	58
3.3.1	Challenges on Definition of Working Time	58
3.3.2	Measurement Strategy Design	59
3.4	Training Data Collection	60
3.4.1	Web Browser Extension	60
3.4.2	Data Collection Settings	63
3.4.3	Results and Findings	63
3.5	TurkScanner	65
3.5.1	System Design	65
3.5.2	Experimental Settings	67
3.5.3	Experimental Results	67
3.6	Conclusion	71
4	CrowdSense: Predictive Model Optimization and Evaluation Based on Subjective Perception of Working Times	73
4.1	Introduction	73
4.2	Related Work	74
4.3	Quantification of Worker Perceptions to Errors in Working Time Prediction	75
4.3.1	Strategy For Estimating JNDs	76

4.3.2	Microtask Survey Design	78
4.3.3	Survey Results	80
4.4	Formulating Perception-Based Functions	81
4.4.1	Evaluation Functions	81
4.4.2	Working Time Range Categorization Functions	83
4.4.3	Objective Function	84
4.5	Perception-Based System Evaluation	85
4.5.1	Settings	86
4.5.2	Experimental Results	87
4.6	Discussion	89
4.7	Conclusion	92
5	Conclusions	93
5.1	Summary of the Dissertation	93
5.2	Summary of Contributions	95
5.3	Limitations and Future Directions	98
5.3.1	Limitations	98
5.3.2	Future Directions	99
	Bibliography	103
	Publications	115

List of Figures

1-1	A list of searched HITs in the Workers’ website of Amazon Mechanical Turk. Each row represents a group of HITs with the same meta information and micro-task contents created with the same requester. Before workers accept and start a HIT, they generally check these types of information to know what they are required to do in the task, as well as how lucrative the HIT would be.	23
1-2	A procedure of how a worker accepts and completes a HIT in Amazon Mechanical Turk.	24
1-3	A comparison between related work and my approach for predicting working times of microtasks. My approach is capable of estimating working time of any microtask posted in a platform, while existing methods makes working time estimates available only for microtasks that were completed by other workers.	29
2-1	Distribution of workers’ total earnings in 2017 (split into 10 groups based on earnings.) I define the top 10%, indicated as “0-10%”, as high-earning extremes — cited from (Kaplan et al., 2018).	43
2-2	A result for the question about online community usage — data has been cited from (Kaplan et al., 2018).	45
2-3	A result for the question about worker tool usage — data has been cited from (Kaplan et al., 2018).	46
2-4	A result for the question “What three TurkoPicon rating(s) do you consider important when selecting a HIT?”	47

2-5	Workers' feelings in terms of time consumption and frustration of microtasks. . .	48
2-6	Criteria for a) HIT selection, b) HIT avoidance / return, and c) ending working session among all workers. While some of the features used to select or avoid HITs are readily available on the platform (<i>e.g.</i> , pay per HIT, Time allotted), others are only available with the use of extensions (<i>e.g.</i> , Requester reputation), and yet others require workers to guess (<i>e.g.</i> , expected completion time, unclear instructions). Error bars represent standard error. Data has been partially cited from (Kaplan et al., 2018).	49
3-1	A data collection procedure. Workers first take our survey HITs to install our browser extension as well as to answer questions about their worker profiles for participating our data collection study. Once the browser extension is installed, it collects data of all HITs visited by the workers together with actual working times.	61
3-2	Interface to record TIME_BTN. (a) The button at the top of the HIT page can be toggled to pause/resume recording working time. (b) A black screen is rendered over the HIT at the beginning as a reminder workers to start the timer.	62
3-3	Working time distribution of microtasks in the dataset (long-tail distribution). . .	64
3-4	The top 30 important features for working time prediction. The importance values were calculated with a split-based measure (by counting numbers of times the feature was used in the model.	68
3-5	Working time prediction results in a confusion matrix, illustrated by a heat map. A large portion of the prediction results are distributed diagonally, which implies that the model successfully captured the trend in the working time prediction. . .	69
3-6	Hourly wage prediction result in confusion matrix shown by a heat map. HIT records with less than ~\$15 actual hourly wage were predicted accurately, while hourly wage of the rest records tend to be predicted as much as they actually are.	70

4-1	Microtask survey interface to evaluate prediction error (of a positive residual) by comparing a predicted working time (left) and the actual working time (right). To evaluate a negative residual, we changed the sentence to “you decided NOT to work” for predicted time and to “someone else ended up spending” for actual time, and we made the actual time shorter than the predicted time.	78
4-2	Survey results for all a_{ij} (blue, white, or red plots), maximal acceptable prediction error e_i in each p_i (black plots), and a curve fitted to the series of $e_i (= \mathbb{E})$ (dashed curve), for the cases of positive and negative residuals respectively. \mathbb{E} was fitted by a log curve (i.e., $f_{pos}(p), f_{neg}(p) = \alpha \log(p + \beta) + \gamma$, where $\alpha, \beta,$ and γ are constants).	82
4-3	Strategy to define ranges based on the fitted function curve of maximal acceptable prediction error.	83
4-4	Psychological amount of working time. The linear function and the log function in gray color are visualized as baseline functions for reference. Offsets ensure that the graph of each function contains the point $(x, y) = (0, 0)$	85
4-5	System performance comparison by working time categories, a) for GBDT and b) for a neural network. In the parenthesis after each actual working time category, the number of tested microtask data whose actual working time is within the range is shown. The working time categories are split based on the evaluation function for positive residual errors, but the accuracy includes both positive and negative errors.	90
5-1	A sample interface for a worker tool that predicts microtask working time based on the proposed technique. The information shown in the dotted box is working time and/or hourly wage of each posted microtask additionally rendered by the tool. The system is capable of providing the information for every microtask regardless of whether or not it was previously completed by other workers.	100

List of Tables

2.1	Description of Mechanical Turk related browser extension tools (as of February 2018) — cited from (Kaplan et al., 2018).	38
2.2	Description of Mechanical Turk related website forums (as of February 2018) — cited from (Kaplan et al., 2018).	39
3.1	List of input features parsed from the collected data. The features consist of three categories and eight sub-categories. The parenthesized numbers in bold text represent the feature dimension sizes.	66
4.1	List of parameter values used for generating comparison pairs ($predicted[s], actual[s] = (p_i, a_{ij})$). For positive residuals, there exist $\sum \mathbb{N}_{pos} = 641$ pairs, wherein $a_{ij} = p_i + jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{pos}, n_i \in \mathbb{N}_{pos})$. For negative residuals, there exist $\sum \mathbb{N}_{neg} = 277$ pairs, wherein $a_{ij} = p_i - jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{neg}, n_i \in \mathbb{N}_{neg})$. Frequencies of p_i, d_i , and n_i were determined based on arbitrary choices made by authors based on the policy of <i>i</i>) successfully determining JND thresholds for each p_i , and <i>ii</i>) sampling adequate data whilst consider as few plots as possible to determine JNDs.	80
4.2	System performance evaluation results based on worker error acceptance. a) Overall accuracy across all tested microtasks; b) Average accuracy for microtasks of which working time was shorter than 510 s (<i>i.e.</i> , the first four working time categories) or longer (<i>i.e.</i> , the last five working time categories.)	88

1

Introduction

1.1 Context of Studies

1.1.1 Definition and Use of Crowdsourcing

Over a decade, crowdsourcing has quickly expanded in its demand (Harris and Krueger, 2015; Kuek et al., 2015). Crowdsourcing, or crowd work, is referred to as a process or a system where certain tasks are outsourced to humans, whose idea was first proposed by Howe in 2006 (Howe, 2006). The term “crowdsourcing” itself is a large term that branches into *paid* crowdsourcing (*e.g.*, competition-based (Tang et al., 2011; Wang, 2002), microtask-based (Ipeirotis, 2010a; Palan and Schitter, 2018)) and *voluntary* crowdsourcing (*e.g.*, wikis (Bryant et al., 2005), citizen science (Raddick et al., 2013; Sullivan et al., 2009; Cooper et al., 2010)). In this dissertation, I focus on *microtask crowdsourcing*, in which small web-based tasks are broadcasted to anonymous users

and completed by them in compensation to monetary reward. In microtask crowdsourcing, a small unit of web-based tasks (**microtasks**) are posted by users called **requesters**, and executed by another type of a large number of anonymous users called **workers**. There currently exist a number of crowdsourcing platforms such as Amazon Mechanical Turk¹, MicroWorkers², and Upwork³ where thousands of microtasks are posted and completed every day.

Crowdsourcing has been effectively utilized for various purposes. Since the beginning of crowdsourcing, microtasks have been created for conducting surveys (Heer and Bostock, 2010; Paolacci et al., 2010) and user studies (Kittur et al., 2008), or for outsourcing creative processes (Neubeling et al., 2016; Kim and Monroy-Hernandez, 2016; Valentine et al., 2017). More recently with a rise in demand for machine learning technologies, humans contributed through microtasks for data labeling (Krishna et al., 2017) and intelligent tasks (on behalf of machines; such as surveillance (Laput et al., 2015; Saito et al., 2016), social conversation (Huang et al., 2017), visual optimization (Bernstein et al., 2011; Koyama et al., 2017), etc.) In every such job domain, crowd markets have made it amazingly easier for practitioners to recruit multiple people online, in comparison to times when most of such jobs could traditionally be done only by people in closed communities. Requesters are now even able to set reasonable wages for microtasks that are done by crowd workers, compared to hiring experts with an extremely high salary. This also means that crowdsourcing is creating more flexible job opportunities for freelancers; crowd workers can work on as many microtasks as they want as long as they are able to find them. More recently, crowdsourcing has also been utilized in the field of machine learning. In response to the growing popularity of Artificial Intelligence, requesters often post data annotation microtasks; crowdsourced annotation is appreciated for its ability to quickly create large datasets for training machine learning models. Furthermore, beyond such *offline* data annotation in batches, there are even “crowd-powered” systems proposed by researchers where crowd workers execute *online* annotation tasks where human outputs are more reliable than machine outputs. Thanks to APIs provided in some crowdsourcing platforms, requesters are able to “embed humans in computers” by automatically posting on-demand microtasks and aggregating workers’ answers.

¹<https://www.mturk.com>

²<https://www.microworkers.com>

³<https://www.upwork.com>

1.1.2 Importance of Microtask Working Time Prediction

In this dissertation, I focused on research in **predicting working times of crowd work micro-tasks**, a challenging problem that has not been fully studied yet. I defined the “working time” in this paper as a duration of time spent on a microtask until it has been completed by a worker. Currently, working time estimation is difficult in crowdsourcing. The main reason is that typical crowdsourcing microtasks are web-based, which are not explicitly associated with a relationship among its HTML contents, a set of interactions expected by the creator to be done by the user, and time spent until completing them. Due to this, it is not always easy for humans to estimate working time simply by looking at the microtask. In addition, working time of a microtask would even be different by who actually does it, considering that workers have many different levels of expertise. For instance, some novice workers would spend thirty minutes on a microtask where expert workers take only ten minutes to complete it with the same quality. Such differences in the expertise of workers stem from many factors such as worker experience years, microtask preferences, and their physical and mental conditions and thus cannot be simply gauged, making it more difficult to estimate the working time.

However, despite the difficulties, I believe that working time prediction is an important technology for us to take several big steps toward the next generation of crowdsourcing. In particular, we would be able to accomplish the followings: *i*) improve worker’s working efficiency by assisting workers locate more lucrative microtasks, *ii*) support creation of generous microtasks by assisting requesters gauge price of their microtasks accurately, and *iii*) build practical crowd-powered systems by enabling real-time control of crowdsourcing.

(i) Worker Assistance. Low worker salary in the current crowd market is one of the difficult situations that makes the sustainability of crowdsourcing under risk. In current crowd platforms, workers usually find microtasks they wish to do from a list of posted microtasks. The primary motivation of workers on crowd work is money (Brewer et al., 2016; Berg, 2015; Martin et al., 2014; Lundgard et al., 2018), thus in most cases workers try to select the most lucrative microtask they can find at the moment. However, locating lucrative microtasks are quite difficult; there are many too-complicated or poorly-paid microtasks created by requesters, but it is not easy to filter them out since workers are given very limited information such as a basic microtask profile (*e.g.*,

a simple textual description, a reward amount, a requester’s name, etc.) and an example of the microtask interface. This is considered to be one of the main reasons why the earnings of workers drop significantly; in fact, a recent study (Hara et al., 2018) reported that an average hourly wage of workers in Amazon Mechanical Turk was merely \$2, which is extremely low compared to \$7.25, the U.S. minimum wage, an ethical minimum wage (Hara et al., 2018; Barowy et al., 2017) even in crowd markets. We need an immediate solution toward this issue in the current crowd market, otherwise the sustainability of crowdsourcing would be constantly under threat. To improve this situation, I believe technology for predicting working times of microtasks would be helpful. By being suggested how long a microtask would take to finish before actually starting it, workers would be able to easily estimate the lucrativeness of the microtask by calculating with its specified reward amount. As an application, for instance, a worker helper tool could be built to visualize estimated working time (and measurement for lucrativeness such as hourly wage) for each listed microtask, thereby enhancing the information given to workers and increasing the transparency of the crowd market.

(ii) Requester Assistance. Related to the above, faulty microtask design and pricing by requesters are also known to be problems that cannot be overlooked. Not all poorly priced microtasks posted in crowd platforms are created by evil requesters (Whiting et al., 2019). Some requesters are novice requesters, just like some of workers, and they often struggle in gauging generous reward amount for their own microtasks (Gaikwad et al., 2017). Even if requesters try to be generous, their microtasks might easily be less reasonable than their expectation by overestimating the execution speed of workers — they sometimes test their microtasks by actually working on it, but they would probably be the fastest to complete it because they are the best persons who know about their microtasks (Hinds, 1999). We know that this actually happens in many cases and is obviously a direct cause that confuses workers in finding lucrative microtasks. To address this, predicting microtask working times would be useful also in assisting requesters for microtask pricing. Again, we could build a tool for application that can properly estimate the working time of a microtask just created by a requester, as well as calculate the hourly wage based on their planned reward amount. The tool could also suggest a difference with a “recommended” ethical reward amount based on the calculation result, considering the worst case where some of them do

not even know the ethical standard pricing.

(iii) Real-Time Crowdsourcing. Working time prediction would also be helpful in controlling the response time of crowd-powered systems. Although a number of crowd-powered interactive applications have been developed today, a remaining common challenge for such interactive systems is controlling the response time (i.e., a sum of worker recruiting time and microtask working time) of workers to ensure the time performance reliability of the system'. However, it is considered to be difficult to achieve due to uncertainty of working time on currently available approaches. Low-latency crowdsourcing is known as one of the possible methods for response time control. There are several approaches for controlling the recruiting time by minimizing it (Bigham et al., 2010; Bernstein et al., 2012; Huang and Bigham, 2017; Haas and Franklin, 2017), the problem of controlling working time problem still remains unclear on them. Although Bolt (Lundgard et al., 2018) has been recently proposed for reducing microtask processing time to milliseconds, it is reported that its domain is still limited for real-world scenarios. On the other hand, if working time prediction is possible, crowd-powered systems could predict its response time, or implement even more effectively than a response time control function. For example, we could build a microtask scheduling system based on real-time computing (RTC) scheduling algorithms (Liu et al., 2000). In such a system, each microtask has its own deadline specified by its requester, and is sorted together with other microtasks in scheduler streams, so that their processing order is properly prioritized to have them completed by workers within the time constraint. Also, requesters could be assisted for reaching agreement on their preferred cost-time balance prior to posting microtasks in back-and-forth communication with the system, by iteratively adjusting time parameters considering suggested cost estimation for the request. This type of crowd-powered system significantly expands the use of crowdsourcing, which would lead to a realization of a more seamless "Crowd-AI" paradigm.

In this dissertation, I will present a series of my research work on predicting working times of microtasks in the context of **(i) Worker Assistance**. Despite that, I believe the same technology can be transferred to other problem domains mentioned above. I thought this was the most urgent problem that needs to be solved, given that crowdsourcing cannot function without the contributions of worker as the members of the workforce. Efforts in preserving better working

environment for workers is essential for the sustainability of crowdsourcing. The primary motivation of workers is known to be monetary reward (Martin et al., 2014). When choosing microtasks, most workers judge whether each microtask is worth completing or not by comparing microtask information and its reward amount set by the requester. Therefore, to maintain the contribution of workers in the market, it is important to ensure that workers can earn reasonably for the work they have done. In the succeeding section, I will describe a detailed background of worker assistance.

Throughout my work in this dissertation I will mainly focus on Amazon Mechanical Turk (AMT) since it is the most popular platform for microtask crowdsourcing all over the world as of 2020 with plenty of rich platform functionalities and external resources such as online communities and worker tools. Although my study was conducted only on a single platform, I believe that all the findings and developed systems presented here shall likewise be applicable for other platforms that are similar to AMT.

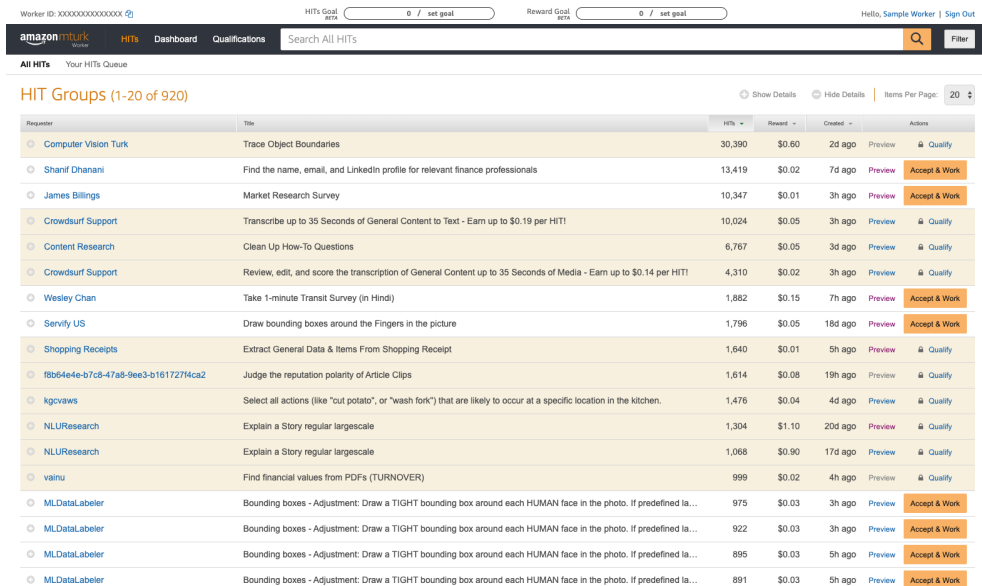
1.2 Background of Crowd Labor and Worker Assistance

In this section, I will first describe how workers typically work in crowd platforms, followed by problems and solutions present in the current status.

1.2.1 Basic Worker Procedure

In AMT, crowd work starts with a list of available *human intelligence tasks* (or **HITs**, which is how microtasks posted in AMT are usually called.) See Figure 1-1 for an interface for the list of all the HIT groups. In this page, workers are shown basic information regarding each listed HIT group, such as a title, a short description, a reward amount, a requester’s name, a time limit duration, days before expiration, and time elapsed since the HIT group was created. Each HIT group holds all HIT instances of the same type (*i.e.*, all HITs that share the same set of values for the basic information parameters); the number of available HIT instances in the group is also provided. Many HIT groups usually have a set of required qualifications, which are granted for determining eligibility to work on the microtask so that requesters can screen out ineligible workers before they start their HITs. There are a lot of different qualification types; there is a “Masters Qualification”

CHAPTER 1. INTRODUCTION



The screenshot shows the Amazon Mechanical Turk interface for viewing HITs. At the top, there are navigation links for 'amazon MTURK HITs', 'Dashboard', and 'Qualifications'. A search bar is present with the text 'Search All HITs'. Below the navigation, the page title is 'All HITs Your HITs Queue'. The main content area is titled 'HIT Groups (1-20 of 920)'. A table lists various HIT groups with the following columns: Requester, Title, HITs, Reward, Created, and Actions. The table contains 20 rows of data, each representing a different HIT group with its own requester, title, and associated metrics.

Requester	Title	HITs	Reward	Created	Actions
Computer Vision Turk	Trace Object Boundaries	30,390	\$0.60	2d ago	Preview, Qualify
Shanif Dhanani	Find the name, email, and LinkedIn profile for relevant finance professionals	13,419	\$0.02	7d ago	Preview, Accept & Work
James Billings	Market Research Survey	10,347	\$0.01	3h ago	Preview, Accept & Work
Crowdsurf Support	Transcribe up to 35 Seconds of General Content to Text - Earn up to \$0.19 per HIT!	10,024	\$0.05	3h ago	Preview, Qualify
Content Research	Clean Up How-To Questions	6,767	\$0.05	3d ago	Preview, Qualify
Crowdsurf Support	Review, edit, and score the transcription of General Content up to 35 Seconds of Media - Earn up to \$0.14 per HIT!	4,310	\$0.02	3h ago	Preview, Qualify
Wesley Chan	Take 1-minute Transit Survey (in Hindi)	1,882	\$0.15	7h ago	Preview, Accept & Work
Servily US	Draw bounding boxes around the Fingers in the picture	1,796	\$0.05	16d ago	Preview, Accept & Work
Shopping Receipts	Extract General Data & Items From Shopping Receipt	1,640	\$0.01	5h ago	Preview, Qualify
18b64e4e-b7c8-47a8-9ee3-b1617274ca2	Judge the reputation polarity of Article Clips	1,614	\$0.08	19h ago	Preview, Qualify
kgcvaws	Select all actions (like "cut potato", or "wash fork") that are likely to occur at a specific location in the kitchen.	1,476	\$0.04	4d ago	Preview, Qualify
NLUResearch	Explain a Story regular largescale	1,304	\$1.10	20d ago	Preview, Qualify
NLUResearch	Explain a Story regular largescale	1,068	\$0.90	17d ago	Preview, Qualify
vainu	Find financial values from PDFs (TURNOVER)	999	\$0.02	4h ago	Preview, Qualify
MLDataLabeler	Bounding boxes - Adjustment: Draw a TIGHT bounding box around each HUMAN face in the photo. If predefined la...	975	\$0.03	3h ago	Preview, Accept & Work
MLDataLabeler	Bounding boxes - Adjustment: Draw a TIGHT bounding box around each HUMAN face in the photo. If predefined la...	922	\$0.03	3h ago	Preview, Accept & Work
MLDataLabeler	Bounding boxes - Adjustment: Draw a TIGHT bounding box around each HUMAN face in the photo. If predefined la...	895	\$0.03	5h ago	Preview, Accept & Work
MLDataLabeler	Bounding boxes - Adjustment: Draw a TIGHT bounding box around each HUMAN face in the photo. If predefined la...	891	\$0.03	5h ago	Preview, Accept & Work

Figure 1-1: A list of searched HITs in the Workers’ website of Amazon Mechanical Turk. Each row represents a group of HITs with the same meta information and microtask contents created with the same requester. Before workers accept and start a HIT, they generally check these types of information to know what they are required to do in the task, as well as how lucrative the HIT would be.

which is given only to a limited number of workers that are approved as experts by AMT, tens of profile-based qualifications (*e.g.*, geographical location, marital/parenthood status, HIT approval rate, HIT return rate, and eligibility for adult content), and a lot of other custom qualifications created by requesters.

Workers then look for a HIT to work and start it. See Figure 1-2 for the basic procedure for accepting and submitting a HIT. From the provided list, a worker selects a HIT that he/she considered starting and opens its web page by clicking a button on the right. Workers are allowed to filter out HITs by keywords, eligibility to work, and a bottom threshold for reward amount. Sorting functions are also available by the number of HITs in the group, reward amount, and creation date. Once workers locate a HIT, they have two options to open the page, either by “preview”ing or by ”preview-and-accept”ing the HIT. When previewing, workers do not yet take the slot for starting the HIT, but they are only shown a sample interface of the HIT, which is usually an initial view of the task; this gives workers a chance to turn down the HIT if they do not like

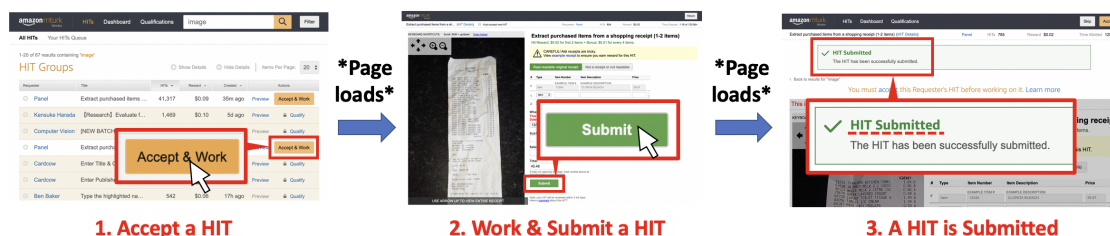


Figure 1-2: A procedure of how a worker accepts and completes a HIT in Amazon Mechanical Turk.

it. On the other hand, by preview-and-accepting (often referred as “Panda”ing) a HIT, workers can directly accept it without previewing, to immediately take the slot so that it is not taken by other workers. Workers still can “return” the HIT if they wish not to complete it and release their slot, but some workers avoid doing this because this would increase their values for the HIT return rate qualification. Once workers start their HITs, they just complete it before its allotted duration expires.

After completing a HIT, workers are allowed to move onto another HIT as much as they wish until they quit crowd work. For workers to receive a reward, they are required to complete the HIT within a pre-specified time limit and to get their answers approved by the requester. During the period wherein they wait for the approval, they keep searching, starting, and completing the next HIT until they consider quitting when they reach their daily goal or if they get tired. As of September 2019, AMT has been providing the beta versions of “HITs Goal” and “Reward Goal” where the platform can count the number of submitted HITs and a total amount of earned reward so that workers can easily compare to each target value set by them beforehand.

1.2.2 Unfair Payment in Crowd Markets

Although such worker procedures in the platform seem quite easy and intuitive, it is actually difficult for workers to earn much because the provided information is not indicative enough, and is relatively scattered. Although there traditionally exists a number of research work that

proposed crowd work design (Kensing and Blomberg, 1998; Moran and Anderson, 1990; Norman and Draper, 1986; Rogers, 1994; Schmidt and Bannon, 1992), it has been reported in a number of research works that many workers are severely underpaid (Gray and Suri, 2019; Horton, 2011; Katz, 2017; , ILO; Durward et al., 2016; Thies et al., 2011). Several studies reported workers typically earned \$2 per hour (Hara et al., 2018; Ipeirotis, 2010b; Shamir and Salomon, 1985). Since \$7.25, the U.S. minimum wage, is the lower limit commonly accepted by researchers of crowd ethics (Hara et al., 2018; Barowy et al., 2017), current worker wage is far from enough. Considering that the main motivation of workers in crowd work is to earn money (Martin et al., 2014), ensuring adequate pay is very important to preserve the system of crowd work. Therefore, further research in worker wage improvement is absolutely necessary, otherwise the current worker environment would easily undermine sustainability of the crowd markets.

A number of previous studies have pointed out that the power imbalance between requesters and workers is one of the main causes of the unfair payment problem (Salehi et al., 2015; Silberman et al., 2010; O’neill and Martin, 2013; Kittur et al., 2013). Requesters are usually given a wide range of discretion in posting microtasks. They are allowed to create microtasks freely; not only are requesters provided with microtask templates to easily create tasks, but they are also allowed to build their own systems and navigate workers to their site for more complex and unique microtasks. Requesters can also set any price for the microtasks that they create. We know that many microtasks are set at very low prices by requesters who do not have much requester experience or who try to save money without consideration for the welfare of the worker. For the created microtasks, requesters can instantly hire their desired numbers of workers whenever needed, with features for screening (Mason and Suri, 2012; Litman et al., 2017), blocking (Karger et al., 2011), and rejecting (Bederson and Quinn, 2011; Wu and Quinn, 2017) workers they do not like.

In contrast, workers are usually provided with very limited functionality in many platforms (Irani and Silberman, 2013; Chilton et al., 2010; Alsayasneh et al., 2017). On major microtask crowdsourcing platforms such as Amazon Mechanical Turk, Prolific⁴, and Microworkers, only basic microtask metadata are made available for workers, such as task prices, requester names, titles, and textual description, as well as simple interfaces just for previewing what the microtask

⁴<https://www.prolific.co>

would look like. To earn efficiently, workers need to immediately judge, based on the provided data, which microtask would provide the best benefit at the moment for maximizing their earnings. Otherwise, they would easily encounter sub-optimal microtasks that require too much time to complete for their suggested prices, making their work routines less efficient. Previous studies in worker ethics have suggested that the U.S. minimum wage should be considered as the lower limit for microtask hourly wage (Barowy et al., 2017; Hara and Bigham, 2017); however, in many cases, workers fail to estimate it because they do not know how to evaluate the given information.

Because of the foregoing, workers often miss many opportunities to earn more because of the power imbalances between requesters and workers, resulting in many workers being paid below minimum wage (Irani and Silberman, 2013; McInnis et al., 2016; Ipeirotis, 2010a; Hitlin, 2016; Horton and Chilton, 2010; Irani and Silberman, 2016; Martin et al., 2014). To address the problem, many researchers have proposed their approaches to assist workers (Chiang et al., 2018; Coetzee et al., 2015; Dontcheva et al., 2014). This astonishing fact clearly emphasizes the need for methods to help crowd workers earn better wages.

Currently, the most practical way for workers to improve their working environment is to utilize third-party resources such as online communities and worker tools. Online communities are websites that provide platforms where users can share information and discuss on anything about their crowd work strategies. For workers, joining the online communities is considered to be important since they can have direct conversations about their questions and thoughts. Worker tools are some sorts of scripts, usually created in a form of a browser extension, a userscript, or a web-based application. Various functions are covered by the worker tools, such as showing requesters' ratings, suggesting newly posted microtasks, and automatically accepting microtasks. In the next subsection, I will introduce the types of online communities and worker tools that are used by AMT workers.

1.2.3 Available Online Communities and Worker Tools

Online Communities

First, I will introduce AMT-relevant online communities. The oldest community for AMT workers and requesters is TurkerNation⁵; TurkerNation started as a forum website, but it has been closed in 2018 and had moved to a Reddit forum (800+ members) as well as a Slack workplace (1,200+ members) since then. Currently, there are no Slack-based platforms for AMT forums other than TurkerNation. Turker Hub⁶ is another forum website for AMT workers built in 2016. Since it had been combined with a new platform called TurkerView mTurk Forum page⁷ in 2018, the forum now accepts posts from requesters as well. TurkerView is becoming very popular among AMT workers (13,500+ members) for its activeness and rich functionality other than that of a forum (introduced below). There also exist MTurk Crowd⁸, Mturk Forum⁹, and Mturkgrind¹⁰ as other AMT-relevant online communities.

Online communities are places where workers (and requesters) can have direct conversations with each other. In most of the mentioned platforms, there are usually discussion threads for categorizing conversations by keywords, such as “General”, “Daily HIT Threads”, “Requesters”, “Scripts & Resources”, etc. These threads are used mainly for posting individual Q&As and discussions; some workers ask for basic worker procedures and better worker strategies, or there are cases where requesters ask workers for hints of creating better microtasks. Through these conversations, workers are able to seek more chances for earning better wages. In all communities, the most active thread is the one related to sharing HITs; in such threads, various HIT information are posted by workers to share which HITs they completed were lucrative and which were not. HITs are often shared with links to the microtask web pages, usually together with Panda links. Some workers even post estimated working time of microtasks, based on how long it took for them to finish it. There are also several Reddit¹¹ threads for AMT workers; From what I found,

⁵<http://turker-nation.com/>

⁶<https://turkerhub.com>

⁷<https://forum.turkerview.com>

⁸<https://mturkcrowd.com>

⁹<https://mturkforum.com>

¹⁰<https://mturkgrind.com> (currently closed)

¹¹<https://reddit.com>

there are “Amazon Mechanical Turk” where workers exchange any sort of information relevant to AMT, “Hits Worth Turking For” where workers share HIT links that workers found were lucrative and/or interesting enough, and “Hits NOT Worth Turking For” where workers share HIT links that workers did not like.

Worker Tools

There are also various kinds of worker tools that are available to AMT workers. It is important for workers to know the reputations of requesters as well as their HITs, so that they can locate and avoid cheaply-paid HITs easily; Turkopticon¹²¹³ is a worker tool powered by a web-based platform where reputations of requesters and their HITs are posted by workers and made available publicly. The score for requester reputation is evaluated with 5-point scales in each of four dimensions of their communicativity, generosity, fairness, and promptness. HIT reputation, which is released in the second version of Turkopticon, includes more indicative and detailed features such as the time required to finish the HIT and the recommendability of the HIT. Its accompanied worker tool is a browser extension that enhances the HITs list view to provide average scores of the accumulated reputation posts for each HIT. MTurk Suite¹⁴ is another browser extension that provides more a sophisticated and integrated view of reputation data in Turkopticon, as well as that in TurkerView¹⁵, another source of requesters reputation posts. Some worker tools are aimed at enhancing HIT search functionalities, since AMT provides only a set of simple filtering and sorting functions. HIT Scraper¹⁶ is a userscript that augments the HIT search interface for filtering searched HITs by Turkopticon ratings and the custom block list of workers and automates refreshing searches through a prespecified interval. Turkmaster¹⁷ is another userscript that adds a side bar to the AMT web page and runs a periodic watcher for HITs in a list of favorited requesters and HITs. There are also worker tools that automate PandaAing HITs; Panda Crazy¹⁸ is one of the most popular userscript that watches HITs posted by saved requesters, and automatically reserves a slot

¹²<https://turkopticon.ucsd.edu>

¹³<https://turkopticon.info>

¹⁴<https://github.com/Kadauchi/mturk-suite>

¹⁵<https://turkerview.com>

¹⁶<https://greasyfork.org/en/scripts/10615-hit-scraper-with-export>

¹⁷<https://greasyfork.org/en/scripts/4771-turkmaster-mturk>

¹⁸<https://greasyfork.org/en/scripts/19168-jr-mturk-panda-crazy>

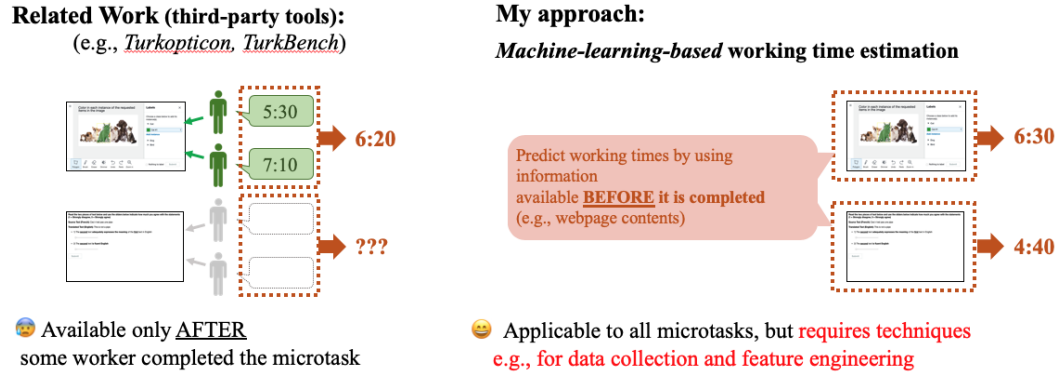


Figure 1-3: A comparison between related work and my approach for predicting working times of microtasks. My approach is capable of estimating working time of any microtask posted in a platform, while existing methods makes working time estimates available only for microtasks that were completed by other workers.

immediately once HITs are posted in the platform. MTurk Engine¹⁹ is a tool that even integrates functions of Panda Crazy functions together with that of HIT Scraper.

1.2.4 Approach

Considering the severe situation of crowdsourcing described above, the purpose of this dissertation is to build a premise of a new working time prediction technique that assists realization of a **fair-trade** crowd market in the future. Currently, several requesters (payers) tend to set excessively low wage to their microtasks, and workers (payees) just have to receive them; such situation highly resembles the labor exploitation problem in labor markets in developing countries. As “fair-trade”, which means that employees pay proper reward to employers, is promoted recently (e.g., SDGs (Organization, 2016),) crowd markets also need to aim for proper microtask pricing. To this end, the first thing that needs to be done is to establish a new technology to predict working time of any microtask, that would let requesters be aware of how the proper pricing is done.

Inferring from what has been developed and utilized by workers until today, working time pre-

¹⁹<https://greasyfork.org/en/scripts/33403-mturk-engine>

diction methods do not yet seem to be fully developed. See Figure 1-3 for a comparison between previous working time prediction techniques and my approach. As introduced above, a few worker tools and communities such as Turkopticon and TurkerBench are only capable of suggesting microtask working times based on information provided by workers. Some workers also shared how long microtasks took for them to complete in the forum websites. However, these methods only allow working time prediction for microtasks that are already completed by at least one worker. This gives the methods some limitations. First, they limit the number of microtasks that can be applied with working time prediction; second, the microtasks that could be applied with working time prediction are also those which are competitive among workers and thus are quickly taken by them, making most of the opportunities unavailable for workers; third, the suggested working times are solely the time spent by the worker who reported it, leaving the differences of worker expertise not considered. By removing these limitations in the current methods, more efficient worker assistance would be possible.

My study explores computational methods for predicting the working times of microtasks based on the past experiences of workers with similar types of microtasks. I built a machine learning-based system that takes various information relevant to a microtask, a worker, and a requester — that could be scraped before starting the microtask — as an input feature vector, and returns a predicted working time in seconds as an output. Such data-driven approach can work very effectively in removing the limitations mentioned above; working times can be predicted immediately once a microtask is posted in the platform even if no worker has worked on it, such that working times of all microtasks posted in a platform can be suggested to workers, and it also takes the profiles and experiences of workers into account for the prediction.

Being the first approach for working time prediction, my approach also involves a number of challenges to overcome. In the following section, I will describe points to be addressed as the main research objectives in this dissertation.

1.3 Research Objectives

There are three different research objectives of my research. These will be identified and discussed in the following subsections.

Investigation of Current Worker Strategy in Tool and Community Usage

First, I tried to emphasize the importance of microtask working time prediction through a survey of worker strategies, particularly focusing on current usage of worker tools and communities. It had been reported in previous literature that workers aided themselves for earning better wage by utilizing worker tools and online communities publicly available (Schmidt, 2015); however, there still existed no research that conducted detailed investigation of such as, which types of tools and communities were popular among workers, why they are widely used, and what are differences in such trends between expert and novice workers. By knowing the tool and community usage of workers prior to building the working time prediction system, we could obtain better insights on what specifically would be the next problem to solve. The first research objective in this dissertation is therefore to formalize workers' knowledge and usage of worker tools and online communities as well as their profiles and daily working strategies, in order to reveal the current status of the crowd work environment. I also intend to explore the investigated results by dividing workers into groups according to their earnings, so that I could be on track with what types of strategies would be actually important for workers to earn more. Based on the analyses, I will conclude with an ideal design policy for a worker tool to develop in my next step.

Data-Driven Approach for Automatic Working Time Prediction

Subsequently, I explored how to design a machine learning-based system that predicts working time. In order to achieve this research objective, there were several challenges to be tackled: The first challenge was defining "working time"; since worker behaviors during microtasks are diverse (*e.g.*, browsing relevant/irrelevant websites in other tabs, taking breaks, and opening multiple microtasks in multiple tabs), there is no single method to calculate working time from the working records of workers. I thereafter attempted to formalize such behavior patterns and design suitable calculation methods to them. The next challenge was data collection for model training. To predict how a worker would spend on a microtask to complete it through a data-driven approach, it was necessary to collect working histories of microtasks to be used as dataset for model training from real workers; however, there seemed to be no easy way for carrying this out. With my definition of working time, I designed a new method for collecting microtask data together with supplementary

data of worker profile- and requester profile-relevant information, both used as parts of an input feature vector for the model, with working time labels annotated in cooperation with the workers. The final challenge I addressed was designing a machine learning model for predicting working times of microtasks. More specifically, I attacked feature engineering problem, to make input feature vectors represent what elements a microtask contains, what type of a requester created it, and what type of a worker is working on it. After an experimental run of the designed regression model for working time prediction, I evaluated the model from several different perspectives such as how accurate the model seemed to predict working times overall, how the prediction results contributed to calculating microtask hourly wages, and what types of features actually helped the prediction.

Quantifying Worker Perception on Working Time Prediction Errors

The final challenge I addressed was optimizing and evaluating the predictive model based on the satisfaction of workers toward prediction results. The aforementioned proposed approach for working time prediction failed to take into consideration what an error presented in each predicted working time would mean to workers. My assumption was that workers would feel differently with regard to the gap between the predicted working time and the time actually spent until completion, and a scale of the working times; for instance, both a prediction error in $(predicted, actual) = (30s, 60s)$ and that in $(1030s, 1060s)$ are “thirty-second differences”, but obviously the former prediction error would be more problematic, whereas that of $(300s, 600s)$ would be more problematic even though it has the same “100% difference” as the first example. Defining such change in workers’ (or humans’) perceptions is not trivial — quantifying such perceptions toward working time would allow us to evaluate and optimize any type of systems that estimate working times of microtasks, which already does exist and are distributed to workers, based on how its prediction results would actually be meaningful to workers. My research objective under this point was therefore empirically defining the worker perceptions of microtask working time, and discussing further the “practical” performance of the proposed model for working time prediction. Based on the metric, I attempted to re-design both the objective function and the evaluation function of the model to discuss the overall prediction performance, which was not

possible to do without the metric.

1.4 Dissertation Organization

For the rest of this dissertation, I present my work for predicting working times of microtasks in a bottom-up order, where its background and pilot market study is described first to build a premise of my work by ensuring that my approach is suitable for the next step toward the status quo of crowd market, followed by explanation of system development for working time prediction.

In **Chapter 2**, I start by reporting results of a survey conducted among 360 AMT workers, aiming to reveal worker strategies in tool and online community usage, prior to building working time prediction system. The survey results indicated that workers who earned more tended to utilize more worker tools and online communities, especially for locating lucrative microtasks or requesters who tend to post lucrative microtasks, although they desired more function on working time prediction. The survey results played important roles in demonstrating strong demands of working time prediction in crowd work environment, thereby emphasizing how essential my work would be for workers.

In **Chapter 3**, I present TurkScanner, a basic concept for building a system that predicts working times of microtasks. I will first explain my definition of working time, considering diversity in the behaviors of workers. Four different types of metrics for working time recording, two automatic and two manual time recording methods, were proposed; each of the four metrics has its unique pros and cons, where I expected that working time would always be recorded fairly accurately by at least one of the proposed metrics. I subsequently explain my approach in collecting data for model training. In order to achieve this, I designed and implemented a browser extension that was to be installed by workers and that collected their working records with data of microtasks completed by them. The browser extension recorded working times of all the four types of the defined metrics, and then asked the worker to select one of them that they felt was the most appropriate based on their actual work, to label the collected data with it. As a result of data collection, I obtained 7303 valid microtask submission records from 83 unique AMT workers who installed the script. Cross-validation was employed for the model evaluation, and all the test results were presented and analyzed in a confusion matrix. Feature importance was also discussed. The

evaluation results indicated that the proposed model was successful in capturing a trend that short microtasks were often predicted as somewhat short and vice-versa, but they also left the problem that it was not possible to discuss at this point how helpful each prediction (with a certain error) would be for workers.

In **Chapter 4**, I present CrowdSense, an empirical definition of the perception of workers with regard to errors in predicting working times of microtasks. To build CrowdSense, I conducted a survey among AMT workers that iteratively asked whether they were able to accept a prediction error presented by a displayed pair of predicted working time and actual working time of a hypothetical microtask. By collecting ~ 100 answers for each of 918 different pairs from 875 unique workers, I obtained log-curve functions that are able to approximate the maximum seconds that general workers would be able to tolerate as a prediction error of a given predicted working time. The obtained curves could be used as evaluation functions, which judged all prediction results whether each of them was within the threshold of workers' tolerance or not, and thus discussed how many percent of all the results would be helpful for workers. Also, objective functions were derived by integrating the curves, which were used to calculate training loss of each prediction error, while considering the workers' perceptions. Evaluation of the new predictive model re-trained under the new objective functions revealed that it was capable of accurately predicting working time across more diverse scales of working time.

In **Chapter 5**, I summarize this dissertation by explaining the future direction of this study.

2

Investigating Worker Strategies and Tool Use Among Crowd Workers

My work originates from a policy of helping workers earn better wages in crowd markets. The main goal for the study described in this chapter is a quantitative analysis on working strategies based on the usage of worker tools and online communities by AMT workers, to get a better sense of what features in the next worker tool would be beneficial for workers. Taking Amazon Mechanical Turk (AMT) — one of the largest crowd work platforms — as an example in this study, I will present the results of the survey I conducted among AMT workers regarding their usual working strategies in AMT and discuss an ideal design of our technologies to support workers.

2.1 Introduction

Various kinds of worker tools and online communities have been developed and utilized among workers in order to make their crowd work more efficient for earning better wages. Most crowd workers are underpaid in current crowd platforms mainly due to power imbalance between requesters and workers; requesters can set any price to microtasks they created, whereas workers are not always provided with enough information to judge value of microtasks and requesters' generosity for efficiently locating potentially-lucrative microtasks. For workers to aid themselves for better microtask selection, there exists various kinds of third-party tools and online communities where workers are able to exchange beneficial information about crowd work with each other. For instance, worker tools are used in obtaining additional information on microtask selection such as requesters' reputations and estimated hourly wage inferred from working histories of workers, or for automatically booking microtask slots as soon as they are posted in the platform. In online communities, workers frequently share links to microtasks they recommend (or do not recommend), have discussions on better worker strategies, or advise requesters for better methods of designing their microtasks. These worker tools and communities are still continuously being developed and actively maintained.

It is important for any kind of tool or community development to get a holistic perspective on their current usage among workers beforehand, just as my study is aimed at building a new method to be implemented in a tool in future. Now that a number of tools and communities have been developed, the platform functionality on workers' side has been significantly enhanced in various ways. However, their details have not yet been investigated in previous literature, in terms of which types of these external resources are especially appreciated, by which clusters of workers they are supported, and why they are valued. In particular, to develop tools that are actually demanded by workers, we should *i*) make a list of currently available tools and online communities, *ii*) examine which features are offered by them, and *iii*) understand which features are widely used and contribute to higher salary. If we could analyze the differences between high-salary and low-salary workers upon these perspectives, I believe the next tool that will significantly help workers would be revealed.

In this study, I seek to better understand the challenges crowd workers face in wage-efficient task selection, and what strategies, tools, and information high-earning workers are using to overcome these obstacles. I conducted a survey on AMT to explore how low- and high-earning workers leverage information on HITs to select tasks to complete and to make inferences about where further research could be best focused to improve the earnings of crowd workers. I examined the task-selection habits and types of external tools utilized by high-earning workers in comparison to their low-earning peers. By investigating these factors, I aim to provide informed design considerations for future tools and task-recommendation systems for improving the earnings of crowd workers.

2.2 Worker Survey Design

2.2.1 Survey Procedure

I created and conducted a survey to gather information about AMT worker earnings and demographics, HIT selection criteria, work strategies, and worker tools. The survey was created and hosted using Qualtrics¹, and 400 HITs including the survey were posted to AMT for workers based in the United States to complete. The survey contained 67 required questions and took between 10 and 30 minutes to complete. Participants were compensated \$3.50 upon completion to provide a mean hourly wage of \$10.

I staggered the release of HITs in order to sample workers with varying levels of crowd work experience as follows: *i*) the first batch of 100 HITs was made available to workers with over 10,000 HITs completed; *ii*) the following three batches of 100 HITs were made available to workers with more than 5000, 1000, and then 100 HITs completed. The survey was limited to workers in the United States and was posted from January 23, 2018 to January 31, 2018.

2.2.2 Survey Questions

The survey began with general demographic questions, including gender, age, employment status, education level, and income. The following survey sections included questions on AMT-related

¹<https://www.qualtrics.com/>

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

Table 2.1: Description of Mechanical Turk related browser extension tools (as of February 2018) — cited from (Kaplan et al., 2018).

Extension name	Description
Turkopticon	A web platform (with API) for reviewing and evaluating requesters and HITs. Also refers to a browser extension that displays pop-ups of the evaluation status on AMT search pages.
Panda Crazy	A userscript that provides an interface for managing and PandA-ing batches of HITs.
MTurk Suite	An extension enhancing AMT pages with features from various scripts and extensions. Includes of Turkopticon, Turkerview, and minor work history and earnings tracking features.
HIT Scraper	A userscript that provides an augmented search interface for HITs. Hit Scraper includes additional search filters and can automation search for new HITs at set intervals.
MTurk Engine	An extension combining HIT Scraper and Panda Crazy features, with an automatic HIT watcher and improved dashboard for managing earnings.
Turkmaster	A userscript that adds a side bar in Mechanical Turk dashboard page. Automatically runs a watcher for new HITs based saved requesters and search keywords. Also supports PandAing HITs.
Greasemonkey/Tampermonkey	Extensions that enable userscripts. (Required for some userscripts, such as HIT Scraper, HITForker, Overwatch, Panda Crazy and Turkmaster)

demographic information, such as time spent working and estimated earnings. Workers were then asked if they had the Masters Qualification on AMT (a “Masters Qualification” is a qualification that is automatically granted to a selection of workers by AMT based on statistical models used to identify workers who “consistently demonstrate a high degree of success in performing a wide range of HITs across a large number of Requesters”².) I also asked if workers felt that the day of the week was a factor in earnings on AMT, and if so, which days were the best and worst for earnings.

Afterwards, I asked the participants about their usage of external resources, AMT related tools (see Table 2.1), and website forums (see Table 2.2), the main focal points of this survey for revealing AMT worker strategies. The participants were first asked how often they utilized tools and website forums when they work. If they answered any option but “never”, they were then shown a set of tools and website forums and asked to answer which type they prefer, respectively. In these questions, an “other” option with a text field in which participants was prepared to provide the participants with additional details. The types of the tools and website forums presented in the survey questions were selected with own discretion, based mainly on the number of users and how

²<https://www.mturk.com/worker/help>

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

Table 2.2: Description of Mechanical Turk related website forums (as of February 2018) — cited from (Kaplan et al., 2018).

Website name	Description
MTurk Crowd (https://www.mturkcrowd.com/)	A community with forum topics such as sharing HIT links, requesters' reputation, scripts/extensions, and AMT news. There are "mentors" for novice workers. 1,130,000+ messages have been posted and 5,200+ members have joined.
Mturk Forum (http://www.mturkforum.com/)	A community with forum topics such as sharing HIT links, requesters' reputation, worker know-hows and habits. The largest platform among our choices; 1,650,000+ messages have been posted and 64,000+ members have joined.
Mturkgrind (http://www.mturkgrind.com)	A community with multiple forum topics such as sharing HIT links and other general discussions. Posts have slowed significantly in the past year. 1,100,000+ messages have been posted and 14,000+ members have joined.
[Reddit] Hits Worth Turking For (https://www.reddit.com/r/HITsWorthTurkingFor/)	A community with a single forum, for sharing good HIT links between workers. 42,000+ members have joined.
[Reddit] Hits NOT Worth Turking For (https://www.reddit.com/r/hNOTwtf/)	A community with a single forum, for warning other workers about bad HITs. 500+ members have joined.
[Reddit] Amazon Mechanical Turk (https://www.reddit.com/r/mturk/)	A community with a single forum, for general conversations/discussions (e.g., various comments on HITs, tips for better tasking, warnings for bad requesters, etc.) 26,000+ members have joined.
Turker Hub (https://turkerhub.com/)	A community with forum topics such as sharing HIT links, scripts/extensions, and wiki information. The newest among our choices; established in Nov. 2016. 559,000+ messages have been posted and 2,200+ members have joined.
Turker Nation (http://turkernation.com/)	A community with multiple forum topics such as sharing HIT links (by workers/requesters) and other general discussions. This forum has 640,000+ posts and 20,000+ members.
HIT Notifier (http://hitnotifier.com/)	Aggregates good HIT links posted on Turker Hub, MTurk Crowd, MTurk Forum, and HITs Worth Turking For and provides an audio notification when new recommended HITs appear.

frequently they were discussed in forums as of January 2018.

I also asked the participants about their various factors taken into consideration in decision making for optimizing their earnings. In particular, the participants were asked about levels of their preferences of each HIT type (*e.g.*, image transcription, survey, external search) with 5-point Likert scales. They were then asked to rate the importance of each suggested factor when selecting HITs, avoiding or returning HITs, and ending their work sessions, also with 5-point Likert scales.

The study was likewise aimed at gauging how much workers would have felt that HITs they work on were frustrating or time-consuming in particular situations during crowd work. The target

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

situations were task search, spending time on returned HITs, and spending time on rejected HITs; Participants were asked to show the level of their subjective feelings with 5-point Likert scales, to gauge how much a HIT was frustrating and time consuming, respectively.

The survey closed with more specific questions regarding the experience and income of workers on Mechanical Turk. Workers were asked to access their AMT dashboard and report the number of HITs approved/rejected/pending, their HIT approval rate, earnings from 2017, and total AMT earnings. These values are available in the AMT dashboard interface, and thus should be more reliable than self-reported estimated wages.

2.2.3 Data Cleaning

Among all the submitted survey answers, I filtered out a part of them through a few spam filtering and validation criteria. After the filtering process, 360 of all 400 survey HITs posted and completed in AMT were kept for analysis. Most of the omitted 40 responses lacked internal consistency; workers' reported approval rate should be consistent with their reported approved HITs divided by reported total HITs submitted. A few submissions that reported zero dollar as total AMT earnings were also skipped. I then manually evaluated the optional open-ended responses to identify obvious spammers (e.g. random strings, repeated questions, consistently unrelated responses). All but two remaining persons completed at least one open-ended meaningful response ("no, none, and nope" were not considered meaningful responses). No additional spammers were identified. All 360 remaining responses reported a HIT acceptance rate within 1% of what would be expected based on their reported HIT submission history, and thus were deemed valid responses.

2.3 Survey Results

In this section, I provide and discuss the results of the survey. I first describe high-level results such as the demographics of the survey respondents, their income levels, and the tools they use. I then perform a more detailed analysis to uncover how and why workers selected particular tasks, the challenges they face, and tools they use. To investigate the effects of external tools and work strategies on the earnings of workers, I split the workers into two groups based on their total

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

reported earnings in 2017 on AMT and compared between groups when relevant. I used total income as opposed to hourly wage as it is available in the AMT dashboard and therefore not prone to estimation errors among reliable respondents. I computed the median 2017 earnings (\$948.18) among the workers who responded to our survey, and thereafter assigned them to the high-earning group if they earn more than the median, and low-earning group otherwise. This resulted in 180 respondents in each group.

I then defined the top 10% of earners in our survey as high-earning extremes and further examine how their habits and strategies differ in comparison to the top 50% of workers. Through these additional comparisons, I aimed at further elucidating successful work strategies.

2.3.1 Demographics

The composition of our survey respondents is similar to the worker demographics reflected in prior research (Ross et al., 2010). Women represented 47.8% of respondents, and the most common age group was 25-34, comprising 39.7% of respondents. More than half of the respondents (61.7%) reported that they are employed full-time, and 50.2% reported having completed a four-year degree or higher. Reported approximate household income (from all sources, including AMT) ranged from “Less than \$10,000” to “Over \$150,000.” The median income bracket was \$40-49,000, and 3% of total respondents reported less than \$10,000.

Reported Earnings.

Self-reported hourly workers’ earnings averaged \$5.12 per hour ($SD = 3.23$) and ranged between \$0.01 and \$25 per hour. Seventeen percent of respondents (62 workers) reported earnings above the current United States federal minimum wage (\$7.25 per hour). Note that given the above details on tracked earnings, hourly reported earning alone may not be an effective means of describing the earnings of workers. Another measure of hourly earnings can be computed per respondent by dividing the daily earnings by average hours worked per day, resulting in a calculated hourly wage. The average calculated hourly wage was \$4.73 ($SD = 3.27$) and ranged between \$0.01 and \$26.67. Given average calculated hourly wage, 16.39% of workers reported earnings above the federal minimum hourly wage.

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

Self-reported daily workers' earnings averaged \$17.3 ($SD = 16.84$) and ranged from \$0.03 to \$100 per day. The low daily earnings may be due to the low hours worked per day. Reported hours worked per day ranged between .5 to 15 hours ($SD = 2.41$) and averaged 3.8 hours per day.

These figures are slightly higher than those reported in previous research works (Ross et al., 2010; Hara et al., 2018). I believe this is due to the staggered distribution of the survey based on the number of HITs a worker has had approved, which resulted in an increase of experienced worker respondents. In fact, individual Spearman non-parametric correlations indicate a positive correlation between experience ($r(360) = .39, p < .001$) and hourly earnings, as well as between experience and daily earnings ($r(360) = .58, p < .001$), thereby suggesting that these figures are slightly inflated due to the sampling method that I employed.

Masters Qualification

Thirty-seven (10.28%) workers reported they had the Masters Qualification. The majority, 28 (75.68%), of them were in the high-earning group. A chi-square test of independence was performed to examine the relation between earnings group (high- vs. low-earning) and Masters Qualification status (with vs. without Masters Qualification). This was significant, ($\chi^2(1) = 10.87, p < .01$). High-earning workers were more likely to have Masters Qualification than low-earners. This may be due to increased access to wage-efficient tasks among those with Masters Qualification. Workers with Masters Qualification reported working an average of just under 2.5 years on AMT prior to achieving the qualification. This time period ranged from one to five years of work on AMT.

User Groups By Earning

See Figure 2-1 for total earnings in 2017, reported by workers as displayed in their dashboard page, represented in percentile groups where the earnings are sorted in ascending order from left to right. The graph demonstrates that the worker earnings increase exponentially. While the total earning amounts of the bottom 70% are barely variant and are under \$2,500, the latter group were incomparably higher and more variant; the top 10% was especially prominent, ranging from \$8,500 to \$26,593 ($M = 13,030.29, SD = 4,818.12$). Their estimated hourly wage averaged

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

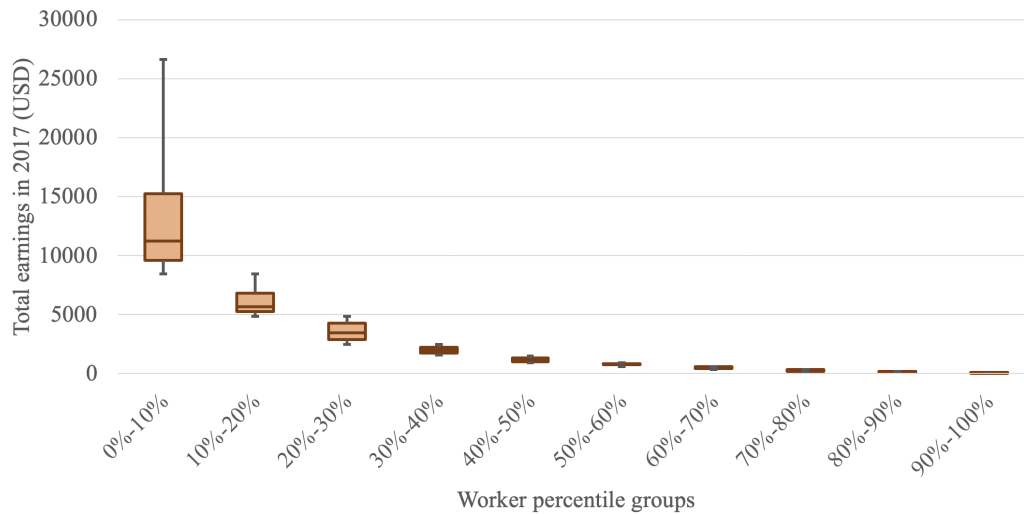


Figure 2-1: Distribution of workers’ total earnings in 2017 (split into 10 groups based on earnings.) I define the top 10%, indicated as “0-10%”, as high-earning extremes — cited from (Kaplan et al., 2018).

\$46.81 and varied between \$20 and \$100 ($SD = 23.27$).

For the rest of my analysis in this chapter, I categorized the participants in accordance with their earnings in 2017 to examine different data trends across the worker categories. I split workers into three categories of *i) high-earning extremes*, the top 10% earners comprised of 36 workers, *ii) high-earners*, the next top 40% earners following the high-earning extremes (144 workers), and *iii) low-earners*, the bottom 50% earners (180 workers.)

2.3.2 External Resource Usage

Worker Tool

See Figure 2-3. 213 (59.2%) respondents reported using extensions to aid their work on AMT. The number of extensions used ranged from 0 to 8 and averaged 2.2 ($SD = 2.24$). Among workers using at least one extension, the average number of extensions used was 3.75. The high-earning extremes were also more likely than high-earners to use browser scripts or extensions when working on AMT ($\chi^2(1) = 11.47, p < .001$), with 91.7% of high-earning extremes using scripts of extensions to augment their work experience. High-earning extremes also reported using

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

a greater number of extensions ($M = 4.11$, $SD = 1.96$) compared to high-earning workers ($M = 2.42$, $SD = 2.42$) ($Z = 3.48$, $p < .001$). The most commonly used extensions were Tampermonkey, Turkoption, and MTurk Suite. "Other" extensions included HITForker (12), Turkerview (4), Overwatch (4), HIT Database (4), and Task Archive (4). Note that HITForker, HIT Database, and Overwatch are Greasemonkey scripts. Four high-earning workers also reported using their own custom scripts.

The most popular extensions used among the high-earning extremes were Tampermonkey (77.78%), MTurk Suite (77.78%), Panda Crazy (77.78%) and Turkoption (72.22%). The usage of Panda Crazy is significantly higher among high-earning extreme workers than high-earners ($Z = 5.21$, $p < .001$). High-earning workers were more likely to use scripts such as MTurk Engine and Tampermonkey. A Wilcoxon Rank-Sum Test indicated that high-earners used significantly more extensions, $Mdn = 3$, than low-earners, $Mdn = 0$ ($Z = 4.49$, $p < .0001$).

Social Platform

See Figure 2-2. More than 60% of workers (222 respondents) reported at least occasionally posting or browsing in AMT related online social spaces. The most popular social platform among workers was the MTurk subreddit where 99 of the surveyed workers used the platform, followed by the HITsWorthTurkingFor subreddit with 80 users, MTurk Crowd with 77 users, and Turker Hub with 49 users. "Other" websites included Facebook groups with seven users and the Turkooption website with five users.

MTurk Crowd was significantly more popular among high-earning workers ($Z = 2.44$, $p < .05$). Twenty-seven percent (48) of high-earning workers used MTurk Crowd, in comparison to 16.11% (29) of low-earning workers. Similarly, Turker Hub was more popular among high-earning workers, with 20% (36) high-earning workers using the site, while only 7.2% (13) of low-earning workers ($Z = 3.53$, $p < .001$) used Turker Hub.

MTurk Crowd was even more popular among high-earning extremes. Over 70.22% (26) over the high-earning extreme workers used MTurk Crowd. This was significantly more than the 20% (36) of high-earners who used the site ($Z = 6.86$, $p < .0001$). Turker Hub was also more popular among high-earning extremes ($Z = 4.52$, $p < .0001$). Forty-seven percent (17) of high-earning

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

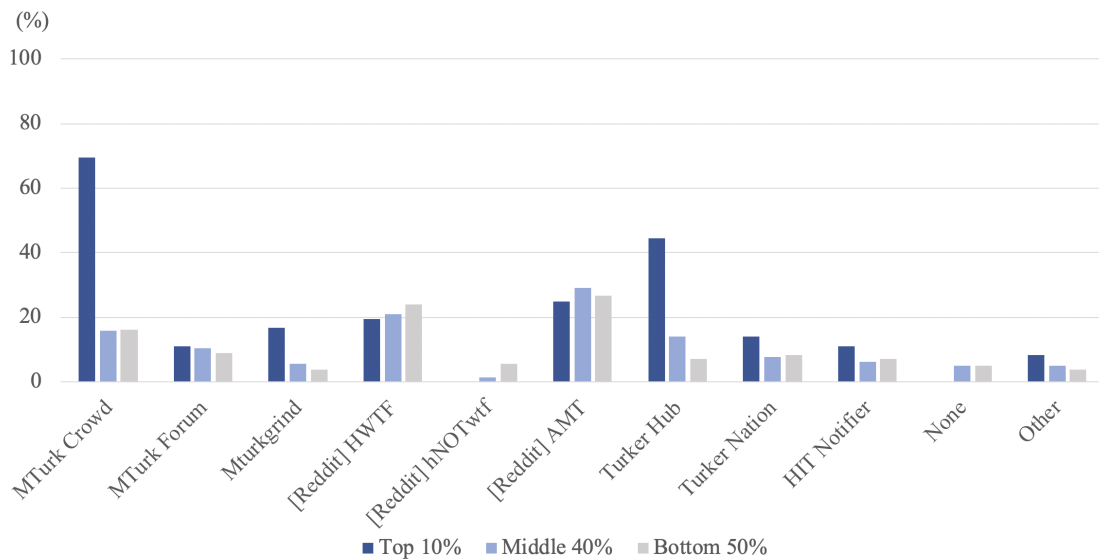


Figure 2-2: A result for the question about online community usage — data has been cited from (Kaplan et al., 2018).

extremes used Turker Hub, in comparison to only 16.11% (29) of high-earners who used the website.

Open-ended responses among high-earning extreme workers also included multiple references to workers tracking their HITs and earnings history and their previous work per requester. In addition, four high-earning extreme workers mentioned a Greasemonkey / Tampermonkey script called MTurk HIT Database which provides this functionality. They were the only workers surveyed who mentioned this script. These responses may indicate the high-earner extreme workers are leveraging information about their previous work to inform current work selection patterns.

2.3.3 Time and Frustration

Task Search

See Figure 2-5 for the answer results of workers’ perception to time consumption and frustration of microtasks. 30% of respondents indicated through a 5-point Likert scale that finding HITs to complete was “4 - Very” or “5 - Extremely” time consuming. Results did not differ significantly between high- and low-earners ($Z = .30, p = .766$). Regarding frustration, 22% of participants (81) reported that task search was “4 - Very” or “5 - Extremely” frustrating.

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

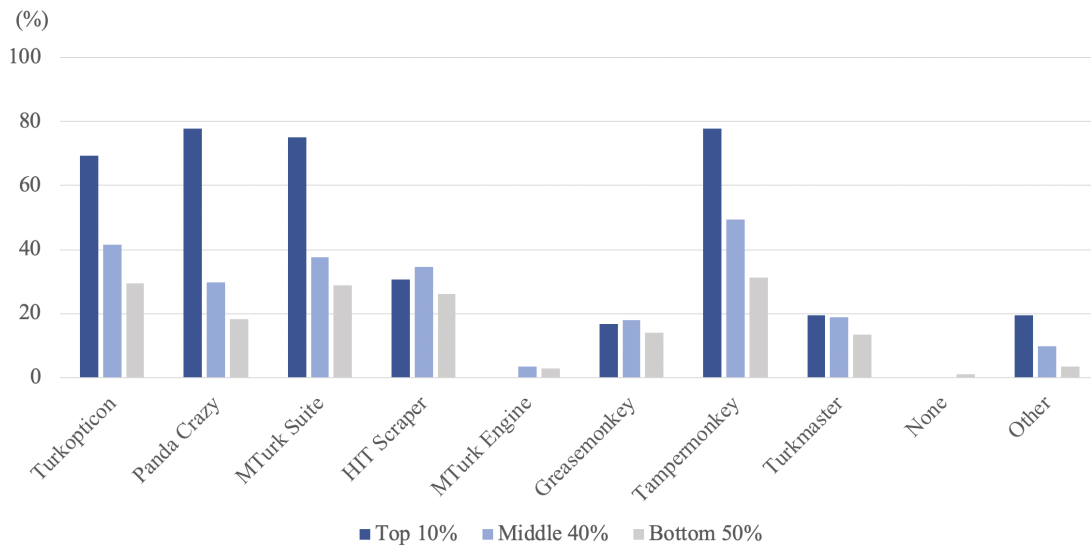


Figure 2-3: A result for the question about worker tool usage — data has been cited from (Kaplan et al., 2018).

Notably, the most important reason for both high- and low-earning workers ending a work session was that workers “Can’t find more HITs worth doing.” Nearly half of participants (48%) indicated that this was a “5 - Extremely Important” motivation in ending a work session. In combination, these findings suggest that the search for HITs on AMT poses challenges for workers of all levels, and improvement to the task search and selection process could potentially improve earnings for all workers.

Rejected / Returned Tasks

Of the total number of respondents, 44% (161) of participants indicated via 5-point Likert scale that having to return a HIT was “4 - Very Time Consuming” or “5 - Extremely Time Consuming.” Similarly, 58% (205) of participants indicated that having to return a HIT was “4 - Very Frustrating” or “5 - Extremely Frustrating.”

Of the total number of respondents, 62% of workers found rejected HITs “Extremely Time Consuming” and 80% of workers indicated they rejected HITs are “Extremely Frustrating.” This means that workers found that Rejected HITs were the most time consuming as well as the most frustrating.

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

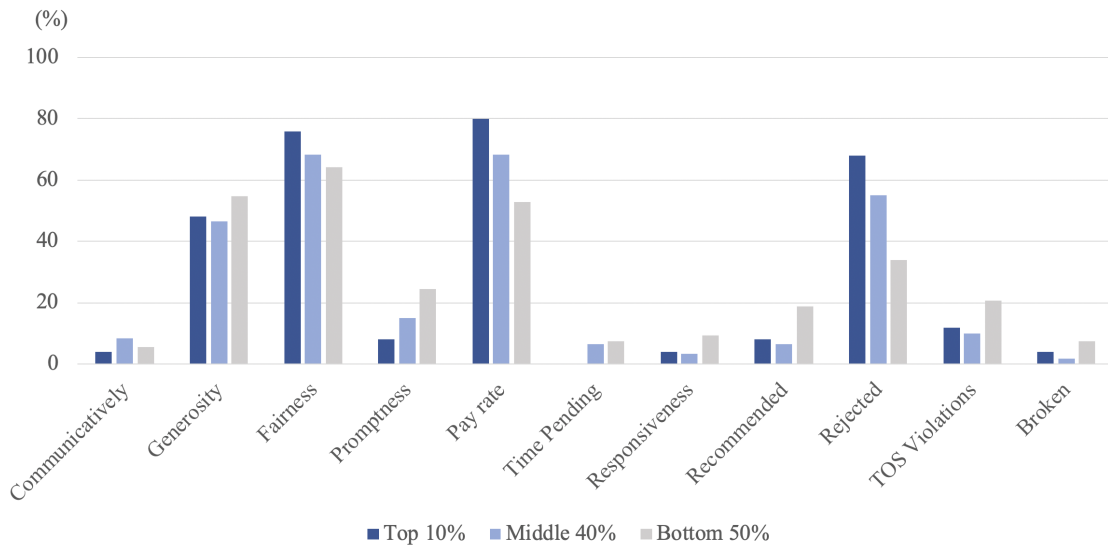


Figure 2-4: A result for the question “What three Turkopticon rating(s) do you consider important when selecting a HIT?”

There were no reliable differences between the high- and low-earning groups in level of frustration ($Z = 1.81, p = .0707$) or reported time consumption ($Z = .43, p = .6670$) for rejected tasks, nor were there any differences in frustration ($Z = -1.04, p = .2975$) or reported time consumption ($Z = -1.48, p = .1380$) for returned tasks.

2.3.4 HIT Type Preference

The most popular HIT types were surveys and extended reading tasks, while the least popular was image transcriptions. High-earners had fewer extreme preferences overall across all HIT types, $M = 2.25$ on a 5-point Likert scale from 1-Not at All Preferred to 5-Extremely Preferred in comparison to the low-earning group, $M = 2.41$. A Wilcoxon Rank-Sum Test indicated low-earners were significantly more likely to prefer surveys ($Z = -4.08, p < .0001$) and image transcriptions ($Z = -2.93, p < .01$) in comparison to high-earners.

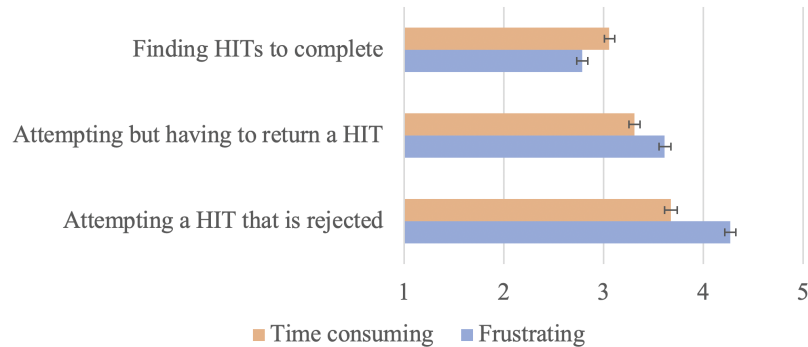


Figure 2-5: Workers’ feelings in terms of time consumption and frustration of microtasks.

2.3.5 Workers’ Decision Making Criteria

HIT Selection

See Figure 2-6(a) for the results of HIT selection criteria. The results demonstrated that the most important HIT selection criteria were “Pay per HIT” whose average score was 4.62, followed by “Expected Task Completion Time” with an average score of 4.14. This indicated that money and time were the most prior factors in selecting HITs for workers. From this result, two things can be implied; first, workers might be concerning the both factors at once, thus pay per time or HIT hourly wage, since it is normal to think that most workers would try to maximize their working efficiency. second, workers would care HIT price and/or HIT completion time individually — some workers would prefer a few HITs with larger pay because they feel much more that they “did” their job, or they would choose many HITs completed in short times because it would lower their risks when their submissions get rejected by requesters (McInnis et al., 2016). “Requester reputation” and “time allotted (to HITs)” follow afterwards; It is reasonable that they are thought to be important, considering the population of users of Turkopticon and TurkerView, but still less probably because these features do not necessarily relate to HITs themselves. Workers indicated that they would care even less about their personal matters such as “Interesting / Fun Task” and “Mental effort”, and about HIT content such as “Media type” and “Input mechanism.”

The least important were “Opportunities to Learn New Skills” and the “Number of HITs Available in a Batch”. The low importance reported for the number of HITs in a batch is surprising,

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

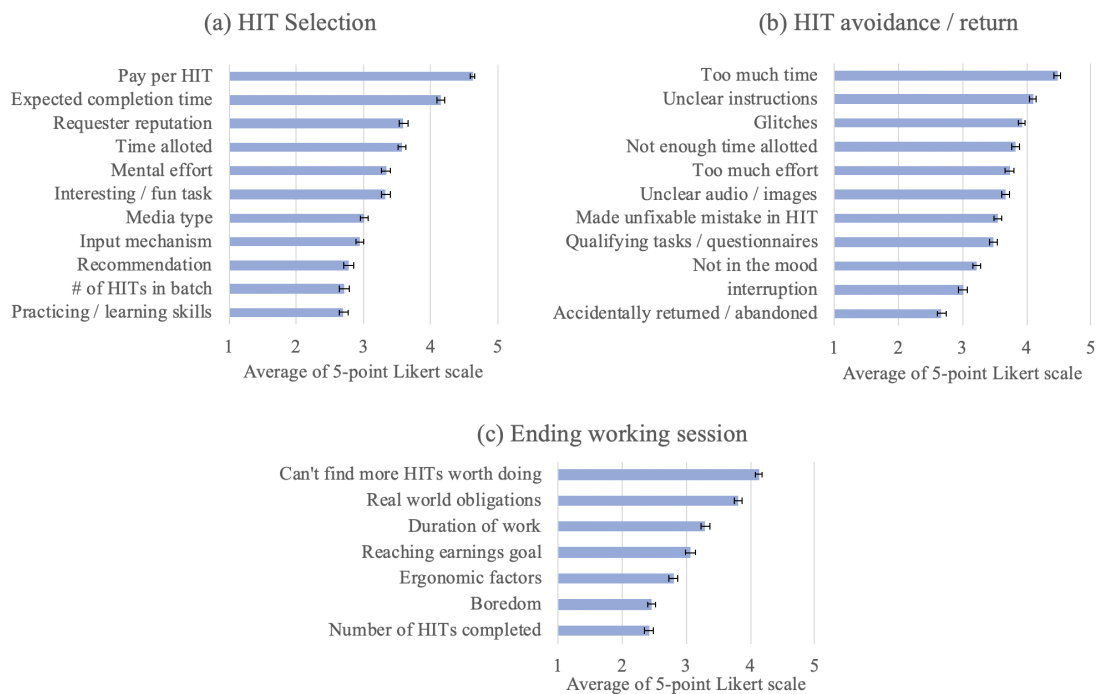


Figure 2-6: Criteria for a) HIT selection, b) HIT avoidance / return, and c) ending working session among all workers. While some of the features used to select or avoid HITs are readily available on the platform (*e.g.*, pay per HIT, Time allotted), others are only available with the use of extensions (*e.g.*, Requester reputation), and yet others require workers to guess (*e.g.*, expected completion time, unclear instructions). Error bars represent standard error. Data has been partially cited from (Kaplan et al., 2018).

given the prevalence of the PandA technique for quickly working through HITs in a batch. In addition, 54 unique respondents (35 in high-earning group and 19 in low-earning group) mentioned working on batches of HITs as part of their work strategy in the open-response questions. Considering the foregoing, I believe that workers are working through batches of HITs, but generally batches are fairly abundant, and batch size is not something that workers must deliberately consider. Instead, in the open-ended responses, workers seemed more concerned about their personal opportunity to seize HITs in a good quality batch. One respondent clarified that, “*I prefer to have something I can work on consistently for a long period of time more than anything, which I’m not sure is answered by any of the above options. It kind of matches “Number of HITs available in batch” but 10000 HITs can be taken in 10 minutes, whereas a batch of 200 might last an*

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

hour.” Seven workers expressed sentiments about how task quality and requester reputation can take precedence over batch size, with users noting that “*when trying a batch with a new requester, I will usually only do 5-10 hits at the most until they approve.*” Others mentioned previewing multiple HITs in the batch before accepting, only accepting batches from a requester they have worked with in the past, or accepting batches only from requesters with high Turkopticon ratings.

The importance of pay rate was evident in task selection based on the open-ended questions. One respondent noted that they, “don’t care what the task is, as long as it pays at least \$12 an hour.” Of the 36 high-earning extremes, 20 included similar sentiments in their open-ended responses.

HIT Avoidance / Return / Abandonment

See Figure 2-6(b) for the results of HIT selection criteria. The results indicated that the most important factor in terms of avoiding and returning HITs are “Task requires too much time for pay” with an average score of 4.5. This demonstrated that workers actually avoid cheaply-paid HITs, also supporting our assumption in the results for HIT selection criteria that workers tend to care completion time and pay of HITs. The following factors consist mainly of problems involved in HIT contents, such as “Unclear instructions”, “Glitches”, “Not enough time allotted”, and “Unclear audio / unrecognizable images.” They relate more to the potential risks for rejection of workers’ submissions or time wasted on the HITs that cannot be completed, rather than regretting cheapness of HITs. Workers do not seem to care their personal matters such as “Not in the mood for this type of task” or “Accidentally returned / abandoned.”

2.4 Discussion

In this section, I will summarize, based on the survey results, what workers were concerned about with regard to earning higher wages as well as what types of assistance are appreciated by them. I will then discuss techniques that would be necessary as a next step of worker assistance by comparing types of worker assistance that are currently available and unavailable.

2.4.1 Available Assistances Appreciated By Workers

According to the survey results, workers relatively did not have much complaints on improving efficiency for *completing* microtasks, but rather they spent more effort on *finding* profitable microtasks. For instance, workers stated that “pay per HIT” and “expected completion time” are more important for their HIT selection criteria, whereas they answered “too much time” and “not enough time allotted” to be dominant considerations for their HIT avoidance criteria. These findings clearly indicate that workers cannot estimate microtasks’ profitability before starting the tasks even though they desire to. Another result revealed that workers’ feelings toward time consumption and frustration of “finding HITs to complete” had the lowest score, while they had larger scores in terms of when the found HITs failed to meet their expectations. On the other hand, workers did not strongly desire to be helped with their task completion by automation techniques of AI. This represents that workers have strong demands for understanding HITs before they actually start them.

In order to solve such problems, workers are currently able to use existing worker tools in collecting auxiliary information of microtasks and automating parts of their microtasks. Forums are more frequently used by high-earning workers, wherein especially active communities such as MTurk Crowd and Turker Hub were referred by them, in order to exchange information on recommendable HITs and requesters. Used worker tools were categorized into two types. Some tools were utilized for understanding microtasks; they estimated what their HITs are about prior to starting them, based on requester reputation provided by Turkopticon (workers particularly concerned about wage-relevant attributes such as “pay rate”, “fairness”, and “rejection”) and (statistically) estimated working times provided by MTurk Suite. Other tools such as Panda Crazy allowed workers to automate the reservation of favorite microtasks, and Tampermonkey to create their own userscript programs to automate completion of particular microtasks.

2.4.2 Further Direction: Working Time Prediction

I believe that technology for working time prediction is essential to allow workers to estimate the required effort without actually seeing HITs. Considering that there were decent workers who supported using Panda Crazy, there is a certain demand for automating PandAing HITs.

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG CROWD WORKERS

On the other hand, it is also known that many of the accepted HITs end up being returned or abandoned because they took longer than expected. All the unpaid times spent on such actions should also be eliminated for improving their wage. Furthermore, the answers of our survey participants regarding the question about Turkopticon rating importance also revealed workers' (especially high-earners) high demand on checking the pay rate of requesters, so that they can estimate preemptively how lucrative each HIT would be in the search list. Therefore, technology that can automatically capture how time-consuming a HIT would be without seeing it — a working time prediction based on microtask contents — is considered to be important.

In addition, working time prediction should be applicable to every HIT that exists in a platform. As mentioned earlier, there is strong demand from workers with regard to how they can quickly judge the lucrativeness of HITs and immediately accept them. Since predicting working times a priori is highly difficult, existing conventional methods such as those conducted in Turkopticon and TurkerView assumes that at least one worker contributes by providing HIT submission records to a system for a HIT and calculates working times and hourly wages based on the contributions, which is a widely appreciated method by AMT workers. However, this method does not allow working time prediction to be applied to HITs as soon as they are posted, and when the information becomes available, the HIT is probably already popular among other workers. In fact, in automation tools for HIT searches, taking estimated hourly wages into account such as those in Turkmaster, Turkopticon-provided hourly wages are unavailable for many HITs. TurkerView has relatively higher availability by calculating hourly wages per requester, but this does not capture hourly wage differences between different HITs posted by the same requester, which would then introduce large errors for some HITs. I assume that academic value would be important in building a new approach that addresses such challenges and in suggesting a tool development strategy that would improve worker earnings.

Working time prediction should also take worker- and requester-relevant information into account. Due to the history-based approach for suggesting estimated working times and hourly wages, all data collected from workers are smoothed out and suggestions no longer consider worker capabilities. It is possible to overcome such problem and assist workers with better microtask selection, by seeking a way for personalized working time suggestion by using workers'

profiles such as worker experience years and how many times the worker had already done the same HIT. Also, as seen in workers' usage of MTurk Suite, workers often want to check which requester posted the HIT prior to starting it. Although directly referring requesters' reputation by themselves is effective in estimating the reliability of the information provided for the HIT, it would be also helpful for workers to utilize requesters' reputation into automatic working time prediction.

2.5 Conclusion

In this chapter, I explored the strategies that low- and high-earning workers use to find and complete tasks. Workers identified pay per HIT as their primary task selection factor and used a variety of worker tools in their attempts at earning higher wages, regardless of their earning level. High-earning workers used more tools and were more involved in worker communities. High earners were also more likely to use batch completion strategies. Through our survey, rejected and returned HITs appeared as key factors in unnecessary unpaid work time and worker frustration.

These findings suggest several avenues of future research in optimizing task selection for improved wages and qualification achievement. Notably, automated task recommendation systems may benefit from collecting HIT content information that allows for automatic feasibility evaluation and work time predictions. Such measures would reduce unpaid work time and improve user access to wage-efficient HITs. I believe these augmentations are likely to improve the overall crowd work experience, and lead to more workers achieving the higher wages that they seek.

CHAPTER 2. INVESTIGATING WORKER STRATEGIES AND TOOL USE AMONG
CROWD WORKERS

3

TurkScanner: Microtask Hourly Wage Prediction

3.1 Introduction

In this chapter, I will explore to build a system for predicting working times of crowd work micro-tasks which was inspired by the result of the worker survey described in the previous chapter. The worker strategy survey had helped me figure out workers' technical needs that they strived to find lucrative microtasks based on requesters' reputations and estimated working times, as well as to book them as quick and many as they could. However, it had been also determined that working time prediction, an essential piece of the technologies to achieve workers' needs, had been realized only with an ad-hoc method, which caused tools' inconvenience due to lack of service availability and evidence of suggestion. This is my main motivation towards building working time prediction

method, and it is aimed for supplementing the lacking piece.

There are a couple of requirements for the working time prediction system studied in this chapter. First, the system needs to be applicable to “any” microtask. Working time prediction based on microtasks, which are comprised of elements that cannot be directly quantified, such as a web page and its metadata, is not easy; due to this, previously proposed methods (Callison-Burch, 2014; Hanrahan et al., 2015) and widely-used worker tools (*e.g.*, Turkopticon, TurkerView) can solely suggest working times of microtasks that are previously completed by other workers, which are simple statistic values calculated based on the working histories. Considering that new microtasks are continuously posted every day, I believe a general method for working time prediction is needed so that it is applicable as soon as microtasks are posted. Second, the system needs to be comprehensively based on features relevant to not only microtasks but also personal information of the actors (*i.e.*, workers and requesters). The aforementioned proposed methods suggest working times by aggregating consequent times spent by workers for completing their microtasks; this may easily give biases of workers’ capabilities to the suggested working times where data is not enough, which would make them unlikely to be the same consequence for other workers with different capabilities. Also, distribution of such biases may be even different by which requester creates the microtasks. In such a case, suggested working times should take this into consideration, but it is not currently possible by the previous methods. Therefore, my proposed system should consider the actors’ characteristics for working time prediction.

In this chapter, I present TurkScanner, a machine learning approach for predicting the working times of microtasks to calculate their hourly wages based on previous logs of other workers for other tasks. This allows workers to judge whether microtasks are worth doing, even for new tasks that no other workers have completed. Below, I will describe a couple of technical challenges to be addressed for building TurkScanner.

The first challenge I addressed in building TurkScanner was to collect reliable ground truth data on the working times of real tasks. Estimating working times automatically is difficult, because I know that worker behavior patterns during microtasks and the motivations behind them are diverse (Kaplan et al., 2018), such as visiting external websites to complete the tasks, taking breaks, or accepting a number of tasks and completing them in a row (Hara et al., 2018). Rather

than attempting to calculate a single working time, I collected three different times (two types of automatic recording and manual recording by the worker), along with the workers' a posteriori judgments on which were likely to be most accurate. The extension collected 9,155 data records of microtask submissions from 84 unique workers. For each worker, I collected information on each of the tasks they completed, including the task (HIT) metadata and HTML content, the reputation of the requester, and the worker profile. I aimed to collect all of the data that workers would view before actually completing a task. Intuitively, I expected results such as "HITs with longer times provided in their metadata may take longer," "HITs posted by requesters with better ratings pay better," and "HITs that included many input elements take longer to complete."

The second challenge was to predict working times (and thus hourly wages) for microtasks using a machine learning-based approach. To the best of my knowledge, TurkScanner represents the first work to utilize machine learning to estimate the working times of microtasks. I extracted ~ 100 features from the data collected from AMT. The cross-validated results showed that TurkScanner achieved hourly wage predictions within a 75% working-time error for 69.6% of all the microtasks (and within a 100% error for 84.3%).

3.2 Related Work

An important piece of information to determine the benefit of a microtask is to know its working time (*i.e.*, how long the microtask would take to finish) before beginning the task. Several researchers and practitioners have explored approaches to estimate working time, aimed at assisting workers to perform better microtask selection (McInnis et al., 2016; Chiang et al., 2018).

Workers often leverage online communities and worker tools for better work efficiency (Kaplan et al., 2018). Among the many existing communities and tools, Turkopticon and TurkerView are major online communities where workers post reputations of requesters and microtasks. In addition, these sites have worker tools that visualize the posted information. Not only do they collect five-grade evaluations of several different reputation criteria and comments from workers, they also ask workers to provide their working times so that they can calculate estimated working time and, thus, hourly wage of the microtask for other workers. However, their methods are history-based methods that only estimate the working time of a limited number of microtasks; it is

only available if the microtask has already been seen and performed by other workers who would presumably contribute their working time to the system. Considering that experienced workers often use other tools that “auto-accept” popular microtasks with high requester ratings such as Panda Crazy, and that new microtasks are posted in platforms on a daily basis, I believe that the current scope for assistance is so small that many workers still remain unaided.

Researchers have also explored several ways to estimate working time. CrowdWorkers (Callison-Burch, 2014) is a browser extension for AMT workers that records user working times on each submitted microtask and calculates its estimated working time and hourly wage for other users. TurkBench (Hanrahan et al., 2015) is a web tool that recommends which microtasks to start next based on their hourly wage and auto-accepts them, so that workers can maximize their hourly wage. It records working times in the background and leverages them for the accurate estimation of an hourly wage. However, again, both tools are based on the working history of workers on specific microtasks. They require many recent records on each microtask, making application of these techniques to previously unseen microtasks difficult.

3.3 Measuring Working Time

TurkScanner estimates a time length required for completing a microtask (“working time”) in a machine-learning-based approach. In this section, I begin with explaining how I gauged the working time in the data collection. I then describe the browser extension design that I used for data collection, followed by an overview of the dataset collection.

3.3.1 Challenges on Definition of Working Time

Gauging accurate working times of microtasks is not easy, owing to the diversity of worker behaviors during microtasks (Bederson and Quinn, 2011). Workers are sometimes asked by requesters to temporarily leave their microtask page and browse external websites or use search engines as a part of their tasks. On the other hand, workers might also be interrupted by browsing other websites irrelevant to the task or leaving from their computers to take a break, or they might accept multiple microtasks and complete them in succession. Therefore, it is necessary to formalize such

behavior patterns to determine whether spent time should be counted as working time, to make automated data labeling possible.

3.3.2 Measurement Strategy Design

In my approach, I designed a heuristic labeling method that records working times in three different ways, each of which has both pros and cons, and ultimately asks workers to choose the most reasonable working time themselves. I expected this method would enable us to label collected data with accurate working times under various conditions. The three methods for recording included:

- **TIME_ALL.** Working time is automatically recorded by a program, from when a worker starts a microtask until when he/she completes it. *Pros:* This is the most reliable method of calculating working time when a worker starts and completes a task without interruption. *Cons:* All of the time spent for irrelevant events (e.g., checking emails, grabbing coffee) is counted.
- **TIME_FOCUS.** Similar to **TIME_ALL**, but only records the time during which the microtask page tab is in focus. *Pros:* This method can exclude time spent for task-irrelevant events. *Cons:* Time spent in other tabs that pertains to the actual task (e.g., survey web pages, Google searches) cannot be properly counted.
- **TIME_BTN.** Workers record working time themselves, by toggling buttons for indicating when they are in the working status. *Pros:* This can cover all worker behavior patterns, even unexpected behavior. *Cons:* This approach fully depends on worker operation, which makes the method vulnerable to human error (e.g., careless or spam response).

For final decisions, I also introduced the following working time type as a fourth option:

- **TIME_CUSTOM.** Manual input for the working time by workers in the case that none of the above three options seem to be correct. *Pros:* This can provide workers with a last-resort option, to label the correct working time when all other recording methods failed. *Cons:* Errors may be present in worker answers.

I set a few hypotheses for judgments of working times for the final labels. First, I expected that `TIME_BTN` would be the most dominant choice; many workers would frequently browse external websites for various reasons and leave from their computers halfway through, and they would be able to record moderately accurate working time by toggling a button. Next, my assumption for the second most dominant choice was `TIME_ALL`. Although this is the most reliable method when workers did not take a break, I knew that some experienced workers often accept multiple microtasks in multiple tabs (Kaplan et al., 2018); thus, I surmised that `TIME_ALL` would not be as frequently used as `TIME_BTN`.

3.4 Training Data Collection

3.4.1 Web Browser Extension

To collect microtask data with working time labels, I recruited AMT workers and asked them to install the web browser extension that scraped microtask data and sent it to the server. See Figure 3-1 for an overview of the data collection procedure with the browser extension¹.

Participants were recruited via an AMT survey microtask. When they accepted the microtask, they were first asked to provide their basic worker profile (e.g., gender, age, household income, worker experience years, weekly working time); then, they were navigated to a web page to install a web script. Workers were advised as they finished installation that they may start the process, contributing to the data collection task for up to 10 days, and they were allowed to uninstall the software at any time. A bonus reward was paid to workers afterwards in proportion to their contribution.

After installation of the browser extension, workers were instructed to work on microtasks (or “HITs” in AMT) as they normally do. Each HIT completion record was collected through the following steps of (a) to (c) in which workers were paid a bonus of 5 cents for completing (b) and (c).

(a) Background data scraping In each HIT page visited by a worker, the web script extracted various microtask-relevant data for input features of the working time estimation system. Fea-

¹<https://github.com/shuwakkumacs/hitscraper/>

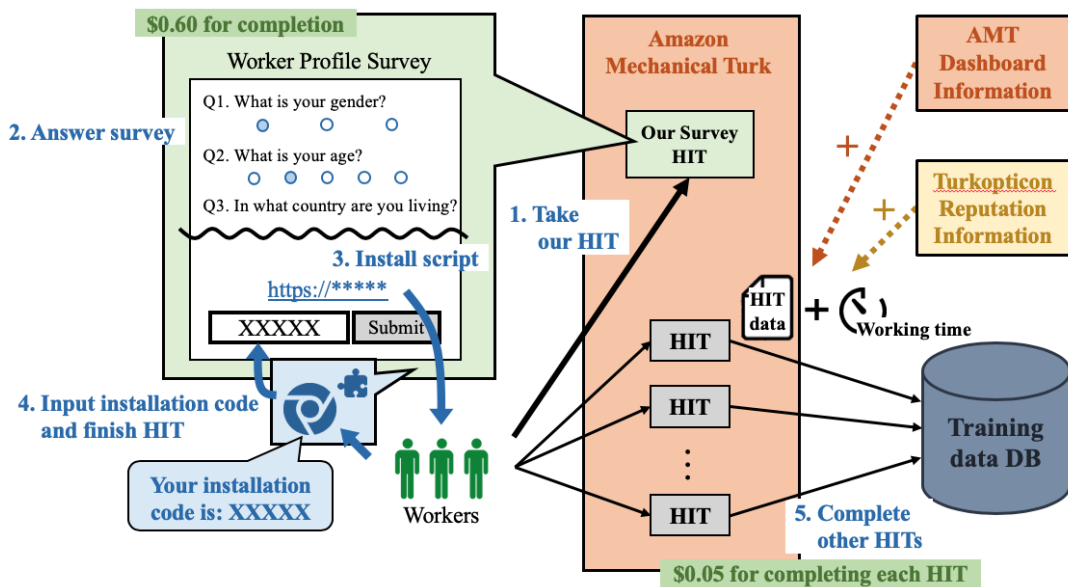


Figure 3-1: A data collection procedure. Workers first take our survey HITs to install our browser extension as well as to answer questions about their worker profiles for participating our data collection study. Once the browser extension is installed, it collects data of all HITs visited by the workers together with actual working times.

tures included microtask information, worker information, and requester information. Microtask information was extracted from HIT metadata and HTML elements in every visited HIT. Worker information was obtained via asynchronous request to the worker dashboard page in AMT once per day, as well as survey responses regarding basic worker information collected before installing the script. Worker features would represent worker capabilities for microtasks (Rzeszotarski and Kittur, 2011), such as their skill levels and learning-curve effects (Yelle, 1979). Requester information is the reputation of the requester of the visited microtask obtained from Turkopticon via provided APIs.

(b) Manual working time recording. As soon as an accepted HIT page was opened, workers were instructed to record working time by themselves until they completed the task. Workers recorded their working states (active or paused) by toggling a button rendered at the top of the HIT interface by the web script (see Figure 3-2.) I implemented two features to prevent workers from forgetting to click their button: First, the HIT page was overlaid with an alert screen when

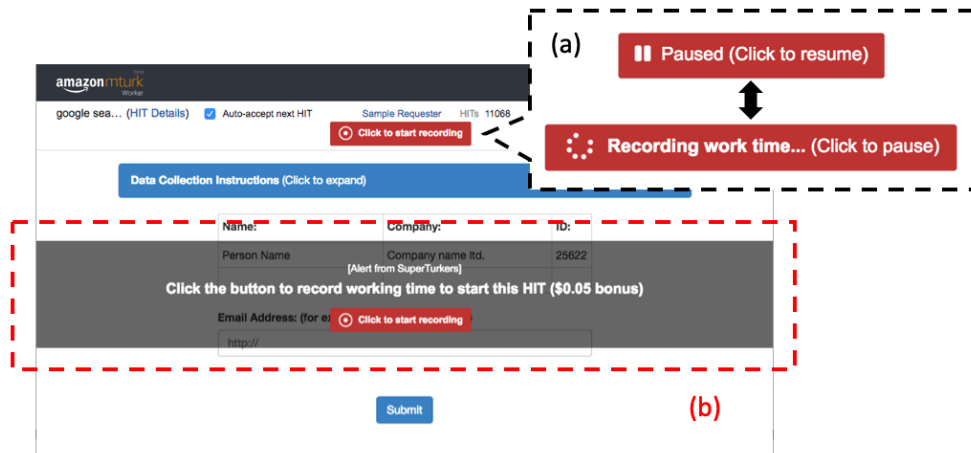


Figure 3-2: Interface to record TIME_BTN. (a) The button at the top of the HIT page can be toggled to pause/resume recording working time. (b) A black screen is rendered over the HIT at the beginning as a reminder workers to start the timer.

HIT was started, such that workers had to click the button to dismiss the alert before starting the HIT. Second, a red border was rendered around the HIT page while the recording button was activated, so that workers could easily be aware of the current status of the button. No more than two buttons were activated simultaneously in multiple tabs, as I assumed that multitasking HITs is not practically possible.

(c) Post-HIT survey. Upon HIT completion, workers selected one recorded working time from multiple choices for labeling the HIT submission record with working time. As soon as a HIT was submitted, a window popped up asking workers to choose one from TIME_ALL, TIME_FOCUS, TIME_BTN, and TIME_CUSTOM as to what they think is the most reasonable answer. When workers judged that none of the first three choices was correct, they selected the last choice (*i.e.*, TIME_CUSTOM) and manually input working time in text boxes (with a “X minutes and Y seconds” format.) After selecting an answer, workers could click a “submit” button to send their answers and dismiss the window.

3.4.2 Data Collection Settings

Worker Recruitment with Pre-Survey

To recruit AMT workers, I prepared a pre-survey that asks for worker profile information (*e.g.*, gender, age, country, household income, worker experience, and worker hours per week). After the survey, I asked workers to install the extension according to instructions and a URL for the installation page. After installation, each participant was given a unique “installation code,” to be copy-and-pasted back into the HIT to verify that they both installed the extension and completed the task. The survey took about 4 min to complete, including the extension installation. I paid workers 0.60 USD (*i.e.*, an expected 9 USD/h) to complete the pre-survey and correctly paste the installation code.

3.4.3 Results and Findings

As a result of the data collection for 10 days in late October 2018, I obtained 7303 valid submitted HIT records, collected by 83 unique workers as participants. The HIT record dataset consisted of 1587 unique HIT groups (batches of HIT instances that share the same microtask metadata and interfaces) created by 977 requesters. On average, participants contributed for 6.5 days ($SD = 3.5$; Median = 8.1) and worked on 109 HITs (Min = 1; Max = 1958; $SD = 238.1$; Median = 34).

Figure 3-3 shows a histogram for working times of the collected HIT records. The working time lengths had a long-tail distribution where a majority of the submitted HITs were completed within a time of shorter than 2.5 min, with a large variance ranging from 3 s to longer than 1 h ($SD = 380.2$ s; Median = 148.3 s; Min = 3 s; Max = 4118 s, Mean = 277.9 s). I believe that a microtask working-time collection resulting in such a long-tail distribution is natural, considering that a majority of workers in crowd markets are beginners (Hara et al., 2018), who likely prefer shorter tasks (Cheng et al., 2015).

For working time labels, `TIME_ALL` was chosen for 3600 (49.3%) HIT records, followed by `TIME_BTN` for 2461 (33.7%) records, `TIME_FOCUS` for 745 (10.2%) records, and `TIME_CUSTOM` for the 497 remaining (6.8%) records. This partly supported my hypotheses, in that `TIME_FOCUS` and `TIME_CUSTOM` were chosen less frequently than the other choices but went against my ex-

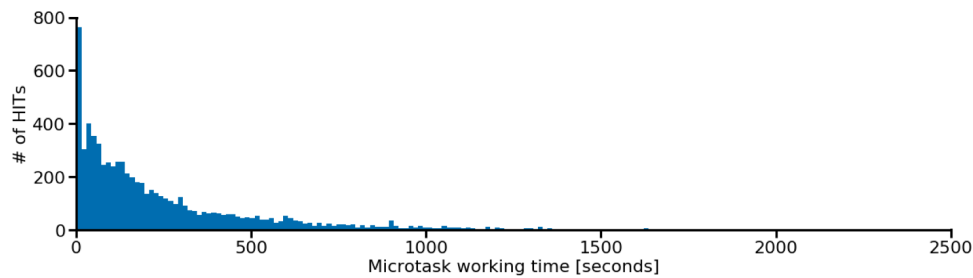


Figure 3-3: Working time distribution of microtasks in the dataset (long-tail distribution).

pectation in that `TIME_ALL` was chosen more frequently than `TIME_CUSTOM` by 15.6 percentage points.

The button was clicked at least once in 6681 (91.5%) HITs. The timer was paused once in 60 HITs, twice in 6 HITs, and four times in 3 HITs, and the timer was never stopped in the remaining 6612 HITs. These results imply that the alert screen seemed to be effective in most cases for promoting workers to activate the recording button, but the red border shown while the timer was activated did not necessarily work in all times. Among the data labeled with `TIME_BTN`, time differences between `TIME_BTN` and `TIME_ALL` were less than 5 s in 75.4% of the records and less than 10 s in 86.9% of the records. This indicates that `TIME_ALL` and `TIME_BTN` recorded almost the same working time in most of the data labeled with `TIME_BTN`, which means that the button was immediately clicked when the tasks were started and were completed without taking a break, but workers still chose `TIME_BTN` likely without any strong reason. However, a rigid evaluation of whether the button was “correctly” used was not possible, because the collected dataset did not contain tracking information of workers during HITs.

Another kind of analysis that would be interesting is to discuss collected working times per HIT type (e.g., image tagging, writing, etc.) However, we would like to leave it for future work, because there is no clear criteria to categorize HIT types, and such categorization would require a lot of hours, which is currently out of our focus in this dissertation.

3.5 TurkScanner

3.5.1 System Design

TurkScanner predicts the hourly wages of microtasks in two steps: 1) estimate the working times of microtasks based on HIT, WKR, and REQ, in a machine learning-based approach; 2) calculate the hourly wage from the rewards and the estimated working times.

Input Features

From the collected data, I extracted microtask-, requester-, and worker-relevant features. See Table 3.1 for the full list of the features.

- *Microtask features* are the most dominant representation among all; they include HIT meta-data (a basic profile of a HIT instance, e.g., time limit, price, HIT batch size) as well as keyword occurrences, URL counts, and input tag counts that were extracted from HTML source code of the microtask. I expected that these features would contribute to the model's representation to learn which kind of microtask contents affect working time and by how much.
- *Requester features* represent the reputation of requesters posted in Turkopticon (Irani and Silberman, 2013) by AMT workers, obtained via API calls. Such information could indicate how appropriate working time might be assumed by the requester, considering the given microtask information.
- *Worker features* include worker profile answered by workers in the beginning of the data collection, as well as a list of AMT worker tools installed in their browser and worker dashboard information. By leveraging such features, the model can possibly consider worker capabilities of crowd work, which would fine-tune the predicted working time of a microtask.

Table 3.1: List of input features parsed from the collected data. The features consist of three categories and eight sub-categories. The parenthesized numbers in bold text represent the feature dimension sizes.

HIT (78) – HIT-relevant information		
META	(5)	HIT metadata set by requesters: time limit, reward, # of HITs in a batch, template type, and HTML text length.
URL	(7)	URL counts in HTML source code: anchor links, in-text links, image links, audio links, video links, Qualtrics ² survey links, and all links.
INP	(36)	Input tag counts and proportion to all: text, submit, radio, checkbox, select, hidden, textarea, number, and other 8 input types.
KW	(30)	Keyword occurrence in either HIT title, HIT description, or a HIT page: "summarize", "survey", "instructions", "opinion", "description", "describe", "read", "click", "audio", "video", and other 20 keywords.
WKR (20) – Worker-relevant information		
PRFL	(8)	Worker profile information collected in pre-surveys: age, country, education level, worker years, daily working hours, weekly working days, etc.
EXT	(8)	Installed AMT-relevant extension tools: CrowdWorkers ³ , Distill ⁴ , Tampermonkey ⁵ , OpenTurk ⁶ , MTurk Suite ⁷ , TurkOpticon, Page Monitor ⁸ , and Auto Refresh ⁹ .
HIST	(4)	Worker dashboard information: # of approved HITs, approval rate, total earnings, and # of HIT submission in a HIT group
REQ (3) – Requester reputation information		
TO	(3)	Turkopticon: average of 5-point scale requester evaluation of generosity/fairness and # of reviews.

Algorithm and Optimization

I predicted the working times of microtasks through regression using gradient boosted decision tree (GBDT) (Friedman, 2001). The model was trained with 101-dimensional feature vectors of task-relevant information (see Section 3.2), by minimizing the mean absolute error between the predicted and actual working times. Upon training and testing, the GBDT model outputs the working times of microtasks on a *log scale* (*base=10*). As shown in Figure 3-3, the working times of microtasks in the dataset admit a long-tail distribution. Taking the logarithm prevents the model from being excessively optimized for short-length microtasks and being affected by outliers.

3.5.2 Experimental Settings

The model evaluation was conducted through four-fold cross validation. When partitioning the dataset, I picked 25% of the 83 workers, and used all their HIT submission records for the test set. This means that the same worker never belonged to both the training and test sets. Therefore, the validation results indicate the extent to which the model is capable of predicting the working time without being trained using HIT submission records of the same worker. To obtain the predicted working times for all the microtasks in the dataset, I tested the model for all the validation pairs, and analyzed them all together. To prevent the results from being too dependent in each trial, I iterated training and testing 50 times, and then calculated the average working time for each HIT record to obtain the results for subsequent analysis.

3.5.3 Experimental Results

In this section, I describe the evaluation results of TurkScanner. I first analyze feature importance, followed by analyses of the prediction accuracy for the working time and hourly wage.

Feature Importance

I first measured the feature importance, to better understand how each feature dimension contributes to predicting the working time. Among the diverse methods available to measure the feature importance, I selected “weight” provided by XGBoost (Chen and Guestrin, 2016), which counts how many times a feature is utilized for splitting across all generated trees. I iterated the training of the initial model 50 times, and then took the averages of the feature importance values for the following analysis.

The important features evenly belonged to all the feature categories of microtask-, worker-, and requester-relevant features. See Figure 3-4 for the ranking list of feature importance. First, the top microtask features included HIT meta data such as reward (1st), time limit duration (3rd), and HTML text length (7th). This is not very surprising since these features are considered as features that would directly affect HIT working time. On the other hand, the top worker-relevant features mainly represented worker experiences such as weekly working hours (4th), the number of approved HITs (5th), the number of total submissions in the same HIT

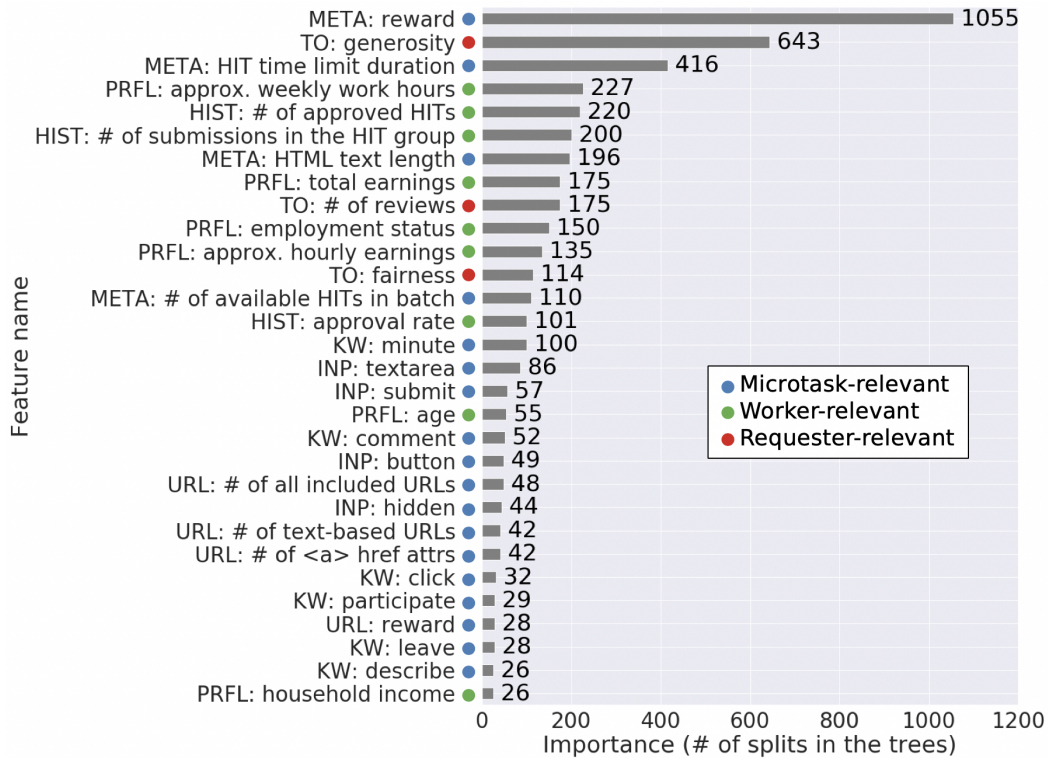


Figure 3-4: The top 30 important features for working time prediction. The importance values were calculated with a split-based measure (by counting numbers of times the feature was used in the model).

group (6th), and worker total earnings (8th). These features are thought to be effective to adjust (although still roughly) estimated working time based on how much workers are good at working on microtasks. Also, the top requester-relevant features (i.e., Turkopticon ratings) were generosity (2nd), the number of reviews (9th), and fairness (12th). While the aforementioned HIT meta data are solely parameters that requesters can change arbitrarily, the requester-relevant features would enable it to control working time estimation by considering their reliability. Other subsequent top features were keywords, URLs, and input tags contained in HITs: for instance, “minute” / “click” / “describe” as keywords, the total number of URLs in the page / URLs that navigate to other pages as URLs, and textarea / button as input tags.

CHAPTER 3. TURKSCANNER: MICROTASK HOURLY WAGE PREDICTION

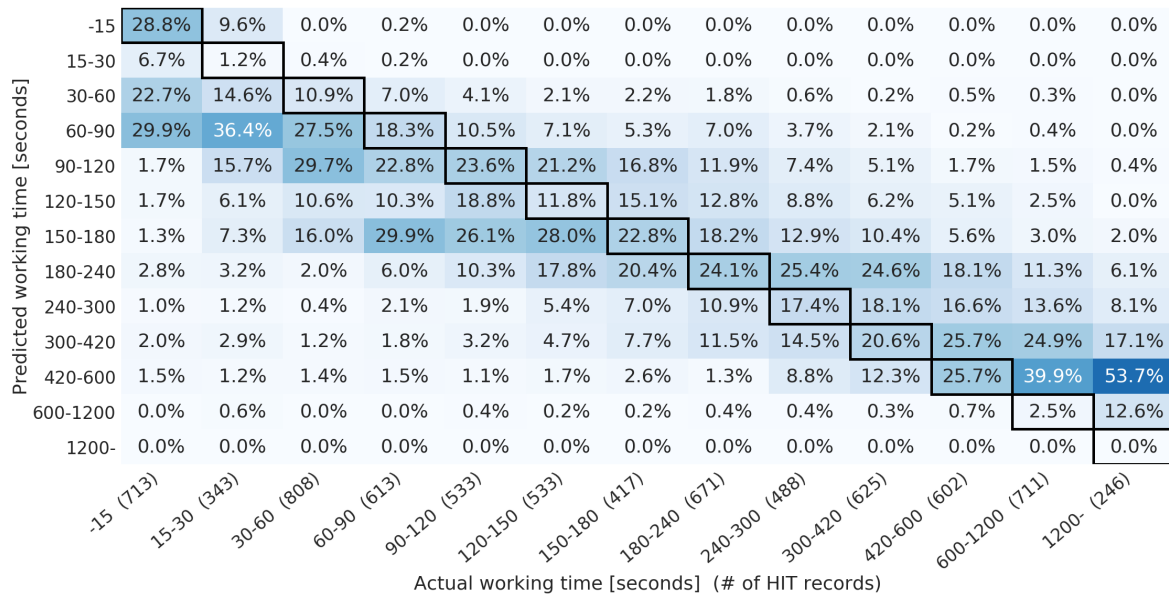


Figure 3-5: Working time prediction results in a confusion matrix, illustrated by a heat map. A large portion of the prediction results are distributed diagonally, which implies that the model successfully captured the trend in the working time prediction.

Per-Working-Time-Range Performance Visualization with Heat Maps

Figure 3-5 presents a heat map of the confusion matrix representing the distribution of the working time prediction results per working time bin. The size of each bin increases gradually towards the right, where the bin of the leftmost columns gathers all the HIT records whose actual working times are between 3 and 8 s, whereas that of the second right-most column includes those between 600 and 1,200 s. Note that I only utilized this binning rule for the analysis: TurkScanner outputs consist of float values for the working time.

The heat map indicates that a large proportion of the predicted working times hit the bin or a nearby bin. Seventeen percent of all the HIT submission records are in the diagonal cells in the heat map, surrounded by bold grid lines, (*i.e.*, they are categorized in the correct bins). Allowing the prediction results to be categorized into neighboring bins, the proportion of correct predictions increases to 47.4% with a one-cell difference (indicated by thin grid lines), and 70.8% with a two-cell difference (indicated by dotted lines). I also note that the predicted working times of HIT records with shorter working time labels (less than 60 s) are likely to be longer. Likewise, the

CHAPTER 3. TURKSCANNER: MICROTASK HOURLY WAGE PREDICTION

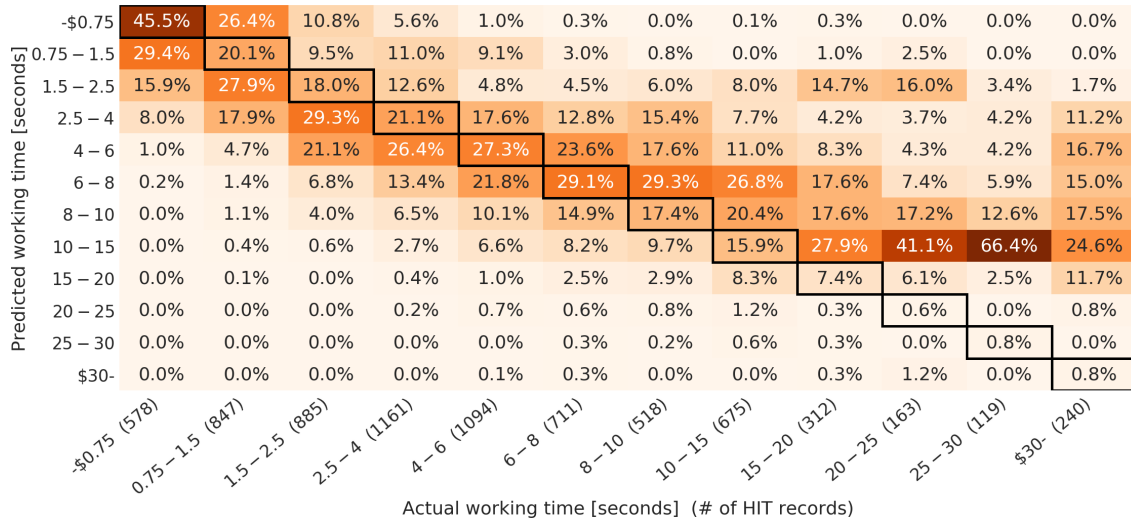


Figure 3-6: Hourly wage prediction result in confusion matrix shown by a heat map. HIT records with less than \sim \$15 actual hourly wage were predicted accurately, while hourly wage of the rest records tend to be predicted as much as they actually are.

predicted working times of HITs with longer working time labels (more than 600 seconds) tend to be shorter. This may be because the distribution of the logarithmic working time labels is closer to the normal distribution (see Figure 3-3b). As mentioned in Section 3.4, the objective function of the GBDT algorithm trains the model such that the overall error across all the data is minimized. Therefore, the model may have been trained to reduce the prediction error for HIT records with medium-length working times, where the largest amount of data samples are.

Hourly Wage Calculation

I calculated the hourly wage using the rewards and predicted working times of microtasks. Over all the tested HIT records, the predicted hourly wage averaged 5.21 USD (SD = 4.53; Median = 4.20). For $N = 5,297$ (69.6% of all the collected HIT records) the hourly wage was predicted within a 75% error, and for $N = 6,412$ (84.3% of all the records) it was predicted within a 100% error.

The prediction performed reasonably well for HIT records with actual hourly wages lower than around 15 USD. On the other hand, many of the HIT records with higher hourly wages were not predicted to be as high as they actually were. To further analyze the incorrect results, I

directly inspected the corresponding HIT records. As a result, I determined that most of these were (i) survey HITs with external URL(s) and (ii) microtasks for which contents were dynamically rendered with JavaScript. These two types of HITs are very similar in that they do not have much static HTML content by themselves. Because I revealed in the feature analysis that some types of HTML content (*e.g.*, text counts in the HIT page, URL counts, and input tags) affected the prediction results, the model might have not been able to predict these HITs accurately.

3.6 Conclusion

In this chapter, I tackled the challenge of predicting the hourly wages of microtasks based on data collected from previous workers. I first presented a data collection method with a web browser extension for gathering data about crowd work and labeling the data with accurate working times. I asked workers to select their answers from choices of working times recorded either automatically and manually by the workers themselves. TurkScanner was then proposed in turn, a system based on the GBDT regression model, to predict working times, and thus calculate hourly wages as its final output. The evaluation methods and results presented in this chapter indicated that TurkScanner would need further design for evaluation and performance improvement. Nonetheless, I clearly showed the possibility that workers can know whether their crowd work will be worth the pay before actually embarking on it, which would make crowd work more transparent and beneficial for workers and requesters.

4

CrowdSense: Predictive Model Optimization and Evaluation Based on Subjective Perception of Working Times

4.1 Introduction

In this chapter, I propose a method to define worker perception for model optimization and evaluation for working time prediction. Since I aim to help workers with formulating working time expectations, my model needs to be optimized and evaluated based on how meaningful the information provided to workers is, and this cannot be determined only through objective values (*i.e.*, seconds). Psychologists have determined that an objective value of some stimulus and human subjective perception to it are often quite different. I assume that this also applies to working

times of microtasks for crowdsourcing. For instance, a prediction error of “30-second difference” for $(predicted, actual) = (30s, 60s)$ would not likely be as acceptable as it would be for $(1030s, 1060s)$, whereas that of “100% difference” in the case of the former would not be as problematic as that for $(1500s, 3000s)$. I therefore believe that any statistic model for time estimation should always be optimized and evaluated according to human perception. Otherwise, worker tools that depend solely on objective measurements could only be evaluated in seconds/percentages for its “overall” accuracy or ‘per-(arbitrary-)category’ accuracy at best, leading to lower satisfaction in user experiences.

To define human sense toward the working time of microtasks, I conducted a subjective survey among AMT workers, randomly and repeatedly suggesting a pair of predicted and actual working times of an imaginary microtask and asked whether the shown prediction error was acceptable. I aggregated 91,060 data samples collected from 875 unique workers to obtain CrowdSense, a set of evaluation results for workers’ perception in whether they accepted a prediction error of a given pair of predicted and actual working times of a hypothetical microtask. CrowdSense enabled us to conclude that the model for working time prediction was capable of predicting $\sim 73\%$ of all tested microtasks in my dataset “accurately”, according to my empirical definition based on worker acceptance toward prediction errors. I also leveraged the CrowdSense for model optimization, and I found CrowdSense-based model optimization enabled accurate prediction of working times across a more diverse range of times than the baseline methods.

4.2 Related Work

Several extant studies focus on investigating the relationship between subjective human perception and objective numerical scales in different domains. As a first in this regard, Weber, in the 19th century, demonstrated the notion of the “just noticeable difference (JND),” which corresponded to the threshold value of a stimulus that humans perceive as a difference. For derivation of his proposed Weber contrast (Fechner et al., 1966), let R denote the strength of a stimulus and ΔR denote the corresponding JND. The value of $\Delta R/R$ always remains constant, regardless of the R value. For example, for an increase in the value of a stimulus from 100 to 110, a corresponding increase from 200 to 220 in the value of the same stimulus is necessary for humans to perceive

an identical change. Subsequently, Fechner derived the Weber–Fechner law (Fechner et al., 1966) by integrating the Weber contrast defined by the relationship $E = C \log R$, where E denotes the subjective perception; R denotes the strength of the stimulus; and C is a constant. The above law is applicable to several phenomena, such as weight, sound, and vision (brightness), and it indicates that humans perceive these phenomena approximately logarithmically. Other studies have also applied the above law in other applications, such as quality of service of communication systems (Reichl et al., 2010) and marketing (Britt, 1975).

In this study, I define the relationship between the subjective perception of workers against microtask completion times and their objective numerical scales. The said relationship has not yet been investigated in extant studies performed in this regard. I believe that development of such a relationship would be a significant contribution to the literature on crowd work and estimation of task-completion times.

4.3 Quantification of Worker Perceptions to Errors in Working Time Prediction

This section introduces CrowdSense — an approach to measure worker perception, i.e., whether or not they “accept” the difference (hereinafter referred to as “prediction error”) between predicted and actual working times. I define workers’ acceptance of a prediction error in the sense that they perceive the prediction error as not being problematic to disturb their workflow.

During model training, my initial concern was to quantify an acceptable accuracy level for working-time prediction. Of course, ideally, a predictive system must not return any error. However, this is not realistic, because there always exist several types of noise that result in prediction of inaccurate working times. Thus, a threshold value for prediction errors must be defined for them to be accepted by workers. For instance, a prediction of $(predicted, actual) = (200s, 250s)$ might be acceptable, whereas that of $(200s, 300s)$ might get rejected. However, when working time have smaller values, both $(20s, 25s)$ and $(20s, 30s)$ might not be problematic but $(20s, 40s)$ might get rejected. Thus, I believed that a minimum prediction error, which workers perceive as unacceptable or problematic must be defined. Such relationships between the objective value of

any stimulus and human perception of the same have been investigated in previous literature (Reichl et al., 2010; Britt, 1975), and the difference threshold is often referred to as the “just notifiable difference (JND)” (Fechner et al., 1966; Sher et al., 2017).

During CrowdSense development, I expected following possibilities. First, CrowdSense would facilitate **evaluation** of system performance based on worker acceptability of prediction errors. For example, the overall prediction accuracy can be calculated by checking whether the pair of predicted and actual working times for each tested microtask was below or above JND (*i.e.*, whether the prediction error would be acceptable to or rejected by workers.) Without CrowdSense, we can only discuss the difference between predicted and actual working times in terms of seconds or the percentage error. This does not explain how meaningful a certain prediction would be to workers. Secondly, CrowdSense would contribute to **optimization** of the predictive model, thereby reducing prediction errors that might be considered problematic by workers. The JND defines, for any given working-time duration, the maximum prediction error acceptable to workers, thereby demonstrating worker sensitivity to the prediction error. Realizing this sensitivity would help one prioritize during model training as to which type of prediction error must be eliminated first, thereby facilitating elimination of problematic prediction errors a priori. However, model optimization without CrowdSense would only allow us to calculate training losses in terms of simple differences between actual and predicted working times. This would not always make the model optimum in terms of worker acceptability of prediction errors; for example, calculating simple differences in seconds would result in relatively larger training losses for long-duration microtasks.

4.3.1 Strategy For Estimating JNDs

For JND estimation, I leveraged the *method of constant stimuli* (Woodworth and Schlosberg, 1954; Kuroda and Hasuo, 2014), considering the human perception in weightlifting as an example. To this end, participants were asked to lift a standard weight followed by a comparison weight. Subsequently, they were asked to judge whether they detected a difference between the two weights. The standard weight is usually fixed to a certain value (*e.g.*, 100 g), whereas the comparison weight is discretely altered within a certain range (*e.g.*, 105 g, 110 g, 115 g, ..., 150 g). Participants were

CHAPTER 4. CROWDSENSE: PREDICTIVE MODEL OPTIMIZATION AND EVALUATION BASED ON SUBJECTIVE PERCEPTION OF WORKING TIMES

asked to compare any one weight pair during each comparison trial. After gathering responses from a sufficient number of participants for each pair, a threshold value of the comparison weight was determined to be that for which more than half of all participants perceived a difference in weight. This threshold was, thus, considered JND with regard to the standard weight.

In this study, the method of constant stimuli was employed to determine JNDs for working times of microtasks. To this end, I designed a microtask that asked workers to compare a pair of suggested working times by considering the predicted and actual working times as the standard and comparison values, respectively. During execution of the survey microtask, workers were instructed to assume a situation wherein they utilized a system that erroneously predicts the working time of a given microtask. Being given a random set of predicted and actual working times for an imaginary microtask, workers were asked whether the difference between the times, or prediction error, was acceptable to them or not. After questioning multiple workers, JNDs were determined for each standard value of the predicted working time by calculating the prediction-error threshold which more than half of all workers perceived as acceptable.

Two different metrics were considered in this study depending on whether the residual (or working-time prediction error) calculated as [actual time]-[predicted time] possessed a positive or negative value. Implications of a positive residual (*i.e.*, $predicted < actual$) are easy to understand; the larger the residual, the more annoyed workers would be, because the system overestimates the benefit, which directly reduces worker earnings. Because this problem occurs when workers actually work on a microtask, I asked workers to imagine that they actually accepted and completed a certain microtask, and then asked if they felt the prediction error was problematic or not.

In contrast, a negative residual value (*i.e.*, $predicted > actual$) necessitates use of a slightly different setting in the survey, since the obtained result is not intuitive. When a worker completes a microtask, the predicted working time of which exceeds the actual working time, the worker can earn more than expected. For this reason, survey with negative residuals would end up collecting more “acceptable” responses for nearly all comparison pairs. However, a system that always predicts working times shorter compared to the actual is not considered a good system. The more the microtasks are undervalued, the more often workers miss opportunities to find lucrative

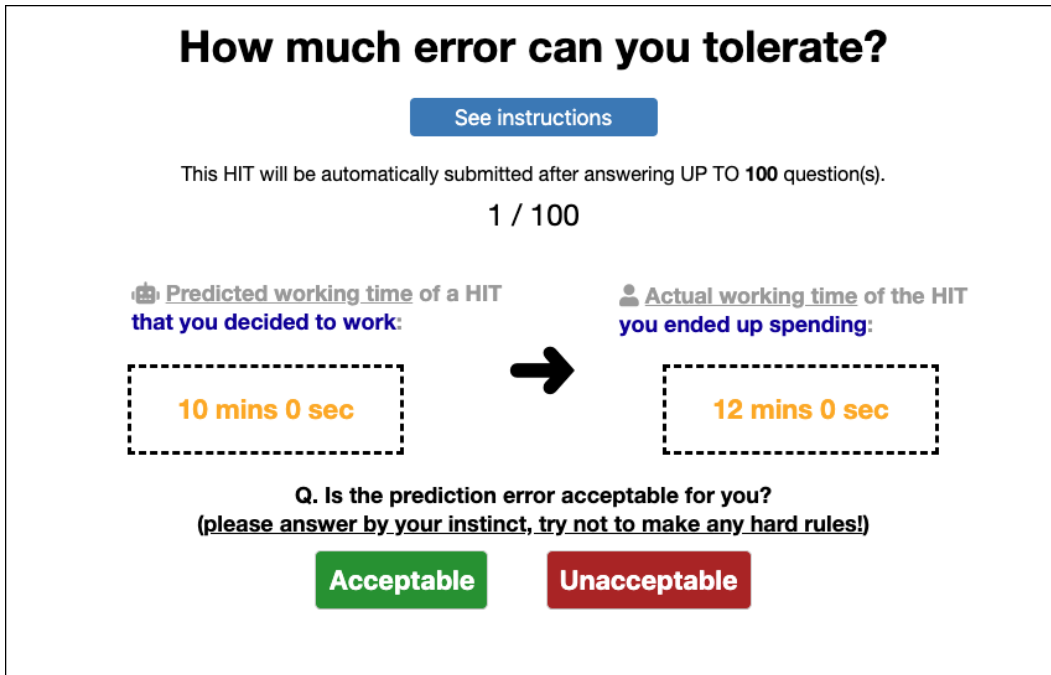


Figure 4-1: Microtask survey interface to evaluate prediction error (of a positive residual) by comparing a predicted working time (left) and the actual working time (right). To evaluate a negative residual, we changed the sentence to “you decided NOT to work” for predicted time and to “someone else ended up spending” for actual time, and we made the actual time shorter than the predicted time.

microtasks. To make workers understand the negative impact of this situation on their potential earnings, the survey instruction was slightly altered; workers were directed to assume a microtask that they decided *not* to do, and that they subsequently knew another worker had completed it faster than predicted, owing to existence of prediction error in the system. Workers were then asked if the incurred error was acceptable to them. By setting the survey question in this way, I expected accurate JND determination in cases involving negative residuals.

4.3.2 Microtask Survey Design

Figure 4-1 depicts microtask interface designs considered in this study. In each pair, workers were shown values of “predicted” and “actual” working times on the left and right, respectively. Whether the calculated prediction error was acceptable to workers or not was recorded by asking

them to click an appropriate response button. The next comparison pair queue was displayed to them as soon as they finished answering the previous question. This sequence continued until the last pair. The microtask was submitted after workers responded to a simple survey that also recorded their comments and feedbacks, if any. Each participant was paid \$1.50 USD (expected \sim \$10/hr) for evaluating up to 100 different comparison pairs (or less when no more pairs were listed in the queue) yielding positive or negative residuals. Each participant was allowed to take surveys for both residual types.

In this study, worker judgments were intentionally recorded by providing with working times exclusively, and no other information was provided, such as the price or type of microtask assigned. It seems to be a natural argument that microtask prices and types must also be provided to survey takers. This is because workers always refer to this information when judging whether a microtask is worth doing, and knowledge of the same might consequently change their response. Although I believe this to be true, it must be noted that there exist both pros and cons in making this information available to workers. While access to this information would provide greater insight into how workers practically evaluate the working time in their daily routines, it is also true that workers often set their own hourly earning goals and microtask-type preferences. Thus, their responses may easily get biased, and results obtained would solely demonstrate demographic distributions of hourly-wage goals and microtask preferences of participants, which are not of interest in this study. It is understood that worker responses vary based on their experience and circumstances. The sole intent of this study was to address this variance by collecting multiple responses from different workers for each sample and averaging them.

In this study, I prepared 641 and 277 comparison pairs with positive and negative residuals, respectively. Subsequently, 19 different time lengths (in seconds) were considered for use as predicted working times varying in the range of 5 s to 1 h, each of which was denoted as $p_i \in \mathbb{P}$. For each p_i , corresponding actual working times were set, denoted by $a_{ij} = p_i + jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{pos}, n_i \in \mathbb{N}_{pos})$ and $a_{ij} = p_i - jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{neg}, n_i \in \mathbb{N}_{neg})$ for positive and negative residuals, respectively, where d_i denotes interval of the difference between predicted and actual working times for each pair. The value of d_i increases upon each iteration of j , and n_i denotes the number of sampled a_{ij} for each p_i . See Table 4.1 for the full list of p_i , d_i , and n_i

Table 4.1: List of parameter values used for generating comparison pairs $(predicted[s], actual[s]) = (p_i, a_{ij})$. For positive residuals, there exist $\sum \mathbb{N}_{pos} = 641$ pairs, wherein $a_{ij} = p_i + jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{pos}, n_i \in \mathbb{N}_{pos})$. For negative residuals, there exist $\sum \mathbb{N}_{neg} = 277$ pairs, wherein $a_{ij} = p_i - jd_i (1 \leq j \leq n_i, d_i \in \mathbb{D}_{neg}, n_i \in \mathbb{N}_{neg})$. Frequencies of p_i , d_i , and n_i were determined based on arbitrary choices made by authors based on the policy of *i*) successfully determining JND thresholds for each p_i , and *ii*) sampling adequate data whilst consider as few plots as possible to determine JNDs.

i	$p_i \in \mathbb{P}$	$d_i \in \mathbb{D}_{pos}$	$n_i \in \mathbb{N}_{pos}$	$d_i \in \mathbb{D}_{neg}$	$n_i \in \mathbb{N}_{neg}$
0	5	5	29	–	–
1	10	10	22	5	1
2	30	10	22	5	5
3	45	10	22	5	8
4	60	10	22	10	5
5	120	15	22	15	7
6	180	10	45	15	11
7	240	10	45	20	11
8	300	10	45	20	14
9	450	20	22	30	12
10	600	20	45	30	15
11	900	20	45	30	15
12	1200	20	45	30	20
13	1500	30	45	30	25
14	1800	30	45	30	30
15	2250	60	30	60	18
16	2700	60	30	60	23
17	3150	60	30	60	27
18	3600	60	30	60	30

values.

4.3.3 Survey Results

In this study, 91,060 worker responses were collected as comparison-pair data samples. The number of participating workers was 875, of which 131 responded to survey questions pertaining to both positive and negative residuals. With regard to positive residuals, evaluations were performed using 60,760 responses provided by 660 unique workers. On average, each comparison pair was evaluated by 95.4 unique workers (median = 97; SD = 8.9; minimum = 62; maximum = 118). For negative residuals, 30,300 comparison responses provided by 346 unique workers were collected.

On average, each pair was evaluated by 109.4 unique workers (median = 110; SD = 5.2; minimum = 95; maximum = 123).

Similarly to what I discussed in Section 3.3, it would have been interesting to analyze the variance of workers' tolerance of prediction error across different microtask types, by specifying what type of microtasks the participants were evaluating. Although I agree that microtask types would make some difference to workers' tolerance, I was not able to conduct such analysis in this study because *i*) there would be a large number of microtask type sub-categories (*e.g.*, image classification and bounding box drawing would vary workers' tolerance, while they belong to the same vision-related microtasks) and *ii*) the results would vary across workers by their preferences and expertise.

4.4 Formulating Perception-Based Functions

4.4.1 Evaluation Functions

Once data collection was completed, an evaluation function was derived based on CrowdSense results. It was expected that the said CrowdSense-based evaluation function would enable us to quantitatively determine the possibility of predicting results with sufficient accuracy so as to be acceptable to workers. This was previously impossible with the exclusive use of objective values of the working time. To this end, I calculated a percentage of “acceptable” votes for each comparison pair to obtain the maximum acceptable prediction error for each p_i , denoted by $e_i \in E$. Using the constant stimuli approach, I defined acceptable prediction errors as those for which the response from 50% or more of all participating workers was recorded as “acceptable.” Subsequently, a series of e_i values were curve-fitted to derive the function $e = f(p)$, where p denotes the predicted working time, and e denotes the residual corresponding to the maximum acceptable prediction error.

See Figure 4-2 for comparison-pair survey results. For the data samples obtained for positive and negative residuals, the least-squares method was employed to fit a function curve in accordance with the relation:

$$e = f(p) = \alpha \log(p + \beta) + \gamma \quad (4.1)$$

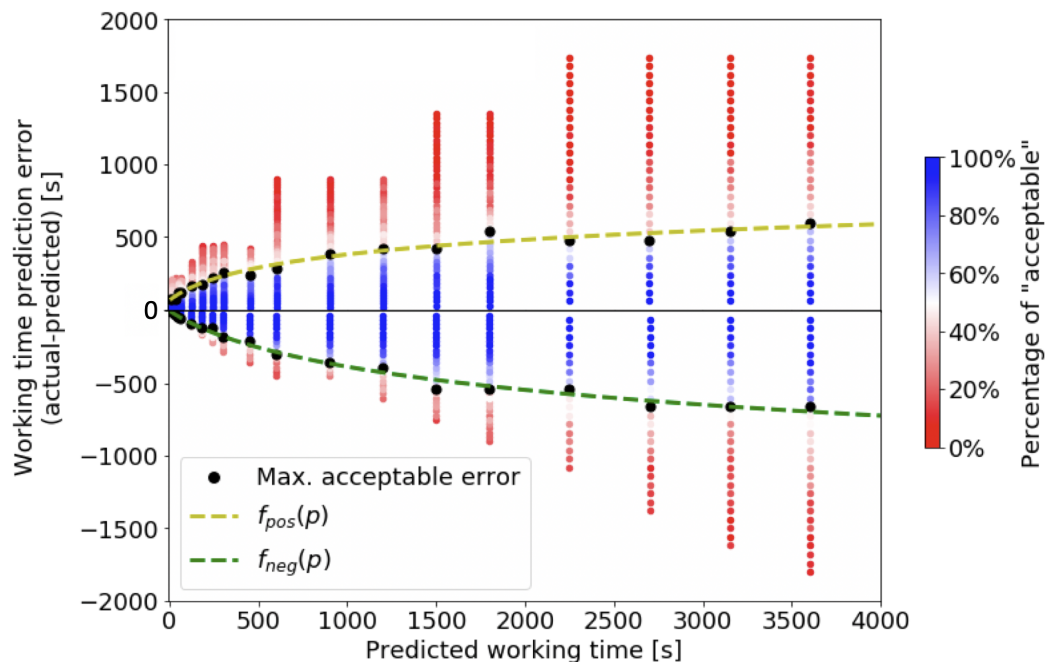


Figure 4-2: Survey results for all a_{ij} (blue, white, or red plots), maximal acceptable prediction error e_i in each p_i (black plots), and a curve fitted to the series of $e_i (= \mathbb{E})$ (dashed curve), for the cases of positive and negative residuals respectively. \mathbb{E} was fitted by a log curve (i.e., $f_{pos}(p), f_{neg}(p) = \alpha \log(p + \beta) + \gamma$, where α, β , and γ are constants).

α, β , and γ are coefficients, calculated under the constraint that the curve function is fitted through all the calculated e_i (the maximal acceptable prediction error for each p_i , shown by black plots.) Using our survey results, values of the coefficients were determined to be $\alpha = 164.3, \beta = 173.1$, and $\gamma = -780.5$ for the positive-residual function ($f_{pos}(p)$); corresponding coefficient values for the negative-residual function ($f_{neg}(p)$) were $\alpha = -289.6, \beta = 358.5$, and $\gamma = 1703.1$. The resulting log-like curve demonstrated an interesting trend indicating that most participating workers were forgiving of large errors with regard to predicted working times for small microtasks, whereas they were likely to accept small relative errors (not exceeding ~ 500 s) for large microtasks up to an hour. In addition, participants were observed to be as tolerant to errors concerning negative-residual comparison pairs as they were to those pertaining to positive-residual ones. However, they demonstrated greater tolerance for prediction errors pertaining to longer predicted working times.

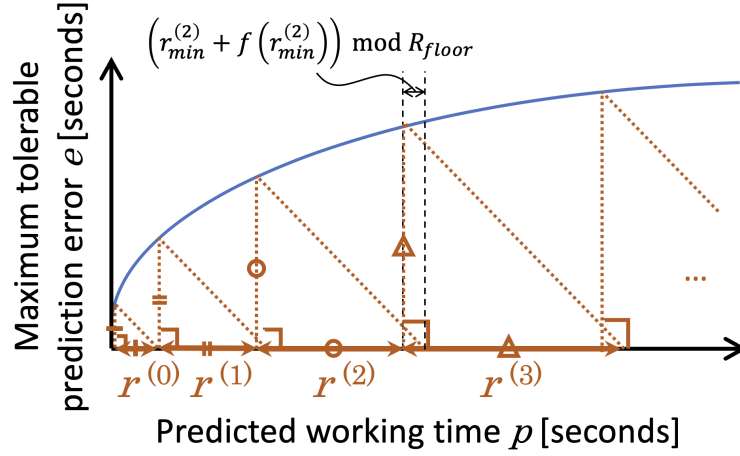


Figure 4-3: Strategy to define ranges based on the fitted function curve of maximal acceptable prediction error.

Based on Equation 4.1, we can then derive a function that calculates a system “accuracy”, meaning how frequent the system is able to predicting working times of microtasks within errors that workers can tolerate. Here we let a set of evaluation functions $f = \{f_{pos}(p), f_{neg}(p)\}$ and an actual working time of a microtask a . If $|a - p| \leq |f(p)|$, the prediction error of the microtask can be regarded as correct as workers would think is not problematic. Thus we define a function for calculating the system’s prediction accuracy, that counts the number of microtasks with acceptable working time prediction error among all n tested microtasks ($T = \{t_0, t_1, \dots, t_k, \dots, t_n\}$) as follows:

$$\text{Prediction Accuracy [\%]} = \frac{\sum_{k=0}^n [|a_k - p_k| \leq |f(p_k)|]}{n} \times 100 \quad (4.2)$$

4.4.2 Working Time Range Categorization Functions

For the evaluation, I determined several regions, each of which represents a time range of prediction errors that workers would accept by using CrowdSense. The ranges of regions were defined as $\mathbb{R} = \{r^{(0)}, r^{(1)}, \dots, r^{(N)}\} = \{[r_{min}^{(0)}, r_{max}^{(0)}], [r_{min}^{(1)}, r_{max}^{(1)}], \dots, [r_{min}^{(N)}, r_{max}^{(N)}]\}$ where $r_{min}^{(k)} = r_{max}^{(k-1)}$, $r_{max}^{(k)} = r_{min}^{(k)} + f(r_{min}^{(k)}) - ((r_{min}^{(k)} + f(r_{min}^{(k)})) \bmod R_{floor})$, $r_{min}^{(0)} = 1$, $r_{max}^{(N)} = \infty$, and $R_{floor} \geq 1$. See Figure 4-3 for a visualization of the region calculations. The calculated re-

gions can be utilized to evaluate the working time prediction algorithm by using a heat map, etc. In this study, I set $R_{floor} = 30[s]$ and $N = 9$ for the parameter values.

4.4.3 Objective Function

I designed objective functions for model optimization based on the two CrowdSense-based evaluation functions. The objective functions were derived with the intention to facilitate calculation of reasonable losses across different working-time ranges based on subjective worker perception. In contrast, as described in Section 1, prediction errors calculated with an objective working-time length would exaggerate losses for large microtasks with longer completion times.

The objective functions were designed by defining the “psychological amount” of working time, as depicted in Figure 4-4. This definition is based on that Weber–Fechner law (Fechner et al., 1966) was used for defining “psychological amount” of any stimuli derived from Weber’s law incorporating JNDs. Reciprocals of evaluation functions f were considered to represent workers’ sensitivity to the prediction error. Accordingly, it was expected that a function obtained by integrating the reciprocal of f would represent a new working-time scale that can also be referred to as the psychological working time (denoted by P):

$$\begin{aligned}
 P &= \{P_{pos}(t), P_{neg}(t)\} \\
 &= \left\{ \int \frac{K}{f_{pos}(t)} dt, - \int \frac{K}{f_{neg}(t)} dt \right\} \\
 &= \left\{ \frac{e^{-\frac{\gamma}{\alpha}} K \cdot \text{Ei}(\log(t + \beta) + \frac{\gamma}{\alpha})}{\alpha}, - \frac{e^{-\frac{\gamma}{\alpha}} K \cdot \text{Ei}(\log(t + \beta) + \frac{\gamma}{\alpha})}{\alpha} \right\}
 \end{aligned} \tag{4.3}$$

where K is a constant and Ei is the exponential integral.

Appropriate prediction losses can simply be defined as the difference between psychological amounts of the predicted and actual working times estimated by using appropriate CrowdSense functions. That is, the psychological amount-based prediction loss can be expressed as

$$\text{Loss} = \begin{cases} P_{pos}(a) - P_{pos}(p) & (a \geq p) \\ P_{neg}(p) - P_{neg}(a) & (a < p) \end{cases} \tag{4.4}$$

where a denotes the actual working time, and p denotes the predicted working time, both measured

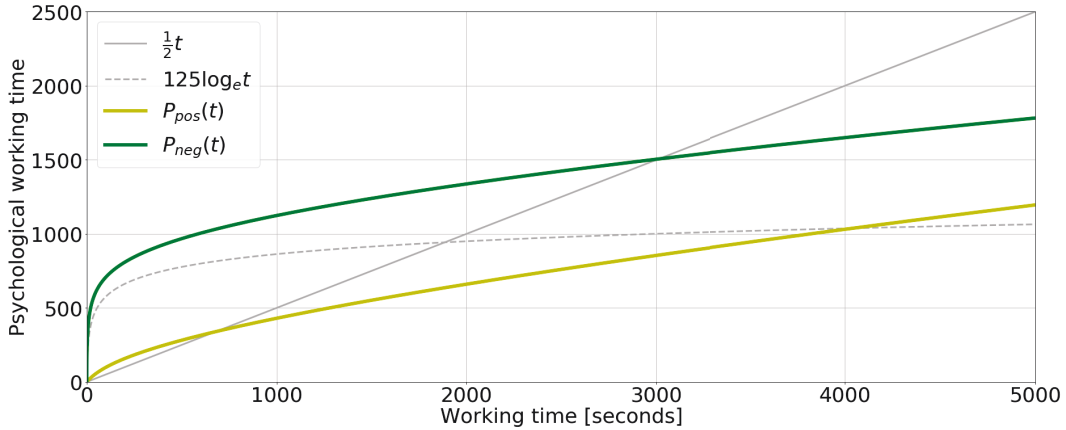


Figure 4-4: Psychological amount of working time. The linear function and the log function in gray color are visualized as baseline functions for reference. Offsets ensure that the graph of each function contains the point $(x, y) = (0, 0)$.

in seconds.

For instance, let us consider calculation of losses on prediction errors in cases where $(predicted, actual) = (30s, 60s)$ and $(1030s, 1060s)$. Because residuals are positive in both cases, the resulting losses can be calculated as $P_{pos}(60) - P_{pos}(30) \approx 25.2$ and $P_{pos}(1060) - P_{pos}(1030) \approx 7.8$, respectively. This demonstrates that penalties can be appropriately calculated, as I mentioned previously that a loss would be smaller when its calculation is based on psychological working time than when it is based on seconds where the working time is longer.

4.5 Perception-Based System Evaluation

In this section demonstrates how CrowdSense facilitates *i)* evaluation of the proposed system for predicting microtask working times based on workers’ perception leading to acceptance or rejection of prediction errors, and *ii)* optimization of the proposed model for more accurate working-time prediction. Defining the “overall accuracy” of a system in terms of the psychological likelihood of workers to accept predicted working times, although with some errors, has not been considered in previous studies performed in this domain (Saito et al., 2019). I hypothesized on the model optimization that CrowdSense would facilitate realization of an all-the-more accurate working-time prediction system capable of operating across different working-time scales

— CrowdSense can define penalties on prediction errors based on workers’ tolerance across different working-time lengths, thereby allowing the predictive model to optimize itself for minimizing the likelihood of “problematic” error prediction regardless of the microtask duration. This study demonstrates the above hypothesis to be true via comparison against baseline methods that define penalties based solely on working-time predictions in seconds or log-seconds.

4.5.1 Settings

Evaluation Method and Criteria: Cross validation was performed with collected datasets described in Section 3. The training and test sets were split in a “task group-open” splitting manner; no microtask from the same group in either the training or test sets was included in the other set. In this study, the CrowdSense-based “system accuracy” was set as the evaluation criterion. For each compared optimization method, prediction accuracy was calculated using Equation 4.2.

Prediction Algorithm: I compared two algorithms to perform regression with Gradient Boosting Decision Tree (GBDT) (Friedman, 2001) and neural network (NN). Regardless of the method employed, a feature vector with 101 dimensions was considered as input, and a scalar value of the predicted working time in s was obtained as the output. In GBDT-based models, LightGBM (Ke et al., 2017) and hand-tuned hyper parameters were used for each compared method, such that its highest overall accuracy was realized upon convergence of its training loss. Values of the maximum tree depth lied in the range of 3–4, and the number of trees (= iterations) ranged between 350 and 450. On the other hand, PyTorch (Ketkar, 2017) was used for building NN-based models that comprised 30- and 10-dimensional hidden layers. Additionally, Rectified Linear Unit (ReLU) (Nair and Hinton, 2010) was employed as the activation function for each hidden layer. Similar to GBDT, hyper parameters for each NN-based model were also hand-tuned. Batch training (batch size = 32) was performed for ~ 1000 epochs to obtain the highest overall accuracy upon convergence of training losses.

Input Features: Using collected data (described in Chapter 3,) microtask-, requester-, and worker-relevant features were extracted. See Table 3.1 containing the comprehensive features list.

Objective Function: For both GBDT and NN models, we compared results obtained using three objective functions that define training losses by calculating the sum of mean squared errors

(MSE) of prediction errors corresponding to *raw*-, *log*-, and *CrowdSense*-scaled working times. The prediction error calculated by using the *raw*-scale working time represents a simple difference in seconds between predicted and actual working times. For example, prediction errors for cases (*predicted*, *actual*) = (60s, 30s) and (300s, 450s) equal 30 and 150, respectively. The *Log*-scaled working time calculates prediction errors based on the difference between logged predicted and actual working times. In other words, prediction errors for the two above-mentioned cases are calculated as $|\log 60 - \log 30| \approx 0.69$ and $|\log 300 - \log 450| \approx 0.41$, respectively. Prediction errors corresponding to *CrowdSense*-scaled working times were calculated by using Equation 4.4. Values of prediction errors for the two above-mentioned cases were calculated to be $P_{neg}(60) - P_{neg}(30) \approx 90.02$ and $P_{pos}(450) - P_{pos}(300) \approx 58.95$, respectively.

4.5.2 Experimental Results

Prediction accuracy by methods: Evaluation results obtained in this study reveal that use of the CrowdSense approach contributes to the determination of both the overall accuracy based on subjective worker perception and best prediction score. Table 4.2a lists overall accuracy values obtained for all compared methods, calculated by using Equation 4.2. As can be realized, in most cases, GBDT-based models demonstrate higher scores compared to NN-based models. The accuracy value of 73.6% was obtained for the GBDT_CrowdSense model. This implies that prediction results for 73.6% of all tested microtasks were sufficiently accurate to be considered acceptable by workers. Scores obtained for other GBDT-based methods were 69.9% (GBDT_log) and 65.3% (GBDT_raw). Results obtained for all NN-based models lied in the range of 60–63%, which are obviously lower compared to those obtained for GBDT-based models.

It should be noted that my intention is not to emphasize that CrowdSense-based optimization scored the best accuracy, but rather to discuss the difference of the accuracy between the CrowdSense-based and objective value-based optimization methods. Since the objective functions and the evaluation functions are both based on the same criteria, it is not very surprising that CrowdSense-based optimization contributed to the best accuracy. However, the prediction accuracy was relatively low when the model is optimized by seconds or log-seconds, which indicates that there is a gap between the workers' perception and objective values of working time. This im-

plies that, without CrowdSense, working time prediction would more likely result in undesirable consequences to workers such as making a worker tool that predicts working times of microtasks intuitively less useful.

Prediction accuracy per working time length: Figure 4-5 illustrates proposed system performance observed for each working-time category. For all comparison methods considered in this study, there was a similar trend that accuracy was higher where working time was shorter. The highest accuracy of $\sim 80\%$ was observed for tested microtasks with under 510 s of completion time. For work time exceeding this value, the accuracy was observed to deteriorate.

Table 4.2: System performance evaluation results based on worker error acceptance. a) Overall accuracy across all tested microtasks; b) Average accuracy for microtasks of which working time was shorter than 510 s (*i.e.*, the first four working time categories) or longer (*i.e.*, the last five working time categories.)

	Accuracy [%]		
	(a)	(b)	
		All	– 510 s
GBDT_raw	65.3	69.9	35.2
GBDT_log	70.4	79.8	9.7
GBDT_CrowdSense	73.6	79.1	31.0
NN_raw	60.5	64.4	27.6
NN_log	63.0	71.3	12.0
NN_CrowdSense	63.5	65.3	39.0

Differences were also observed between trends pertaining to the two baseline optimization methods — log10 and raw working-time-scale-based — when employing both GBDT- and NN-based models. Model optimization employing the log10 working-time scale demonstrated realization of the highest peaks between 60 s and 300 s, and its accuracy became extremely low with increase in working time beyond 510 s. In contrast, the raw-scale-based optimization demonstrated lower scores corresponding to shorter working times. However, it was found to predict longer working times more accurately. These differences support my hypothesis that the log-scale-based

optimization would exaggerate error penalties for shorter microtasks whilst ignore those for longer tasks, and that the raw-scale-based would operate vice-versa. This makes overfitting of the model possible towards the exaggerated side.

However, contrary to the above discussion, the CrowdSense-based optimization was observed to be rather successful in leveraging attributes of both baseline methods. For a simple analysis of the difference between working-time lengths, the working time was divided into two groups comprising four and five regions, respectively, by means of a threshold value of 510 s. Average accuracies for these groups are separately listed in Table 4.2b. In both groups, the use of CrowdSense-based optimization demonstrated the highest accuracies. When employing the GBDT-based regression model, scores obtained using the CrowdSense-based approach demonstrated a difference of -0.7 points for ≤ 510 s and -4.2 points for > 510 s. Corresponding differences observed when employing NN-based regression equaled -9.9 and -27.9 points, respectively. When employing the NN-based model, CrowdSense demonstrated better accuracy compared to the raw working-time-based optimization for the former category, and the best performance for the latter category. This clearly demonstrates that use of the CrowdSense-based approach contributes towards reducing biases across the entire working-time range considered for baseline-optimization methods.

4.6 Discussion

This study explored development of a system for accurately predicting the working time of microtasks by leveraging the subjective perception of workers with regard to the working time. This also resulted in reasonable optimization and evaluation of the underlying prediction model. I will discuss below the contribution of the proposed technique and its possible future directions.

Respecting objective value-based evaluation: As I showed in the experimental results, objective value-based model optimization give biases for penalties of prediction errors across different time scales. By seconds-based optimization, the system was capable of more accurate time prediction for long microtasks but not for short microtasks, and vice-versa by log-seconds-based optimization. Our experimental results demonstrated that the proposed workers' tolerance-based method would be very useful to build a new optimization criteria that takes only the good points of the both objective value-based criteria. In this sense, I believe the workers' tolerance-based

CHAPTER 4. CROWDSENSE: PREDICTIVE MODEL OPTIMIZATION AND EVALUATION BASED ON SUBJECTIVE PERCEPTION OF WORKING TIMES

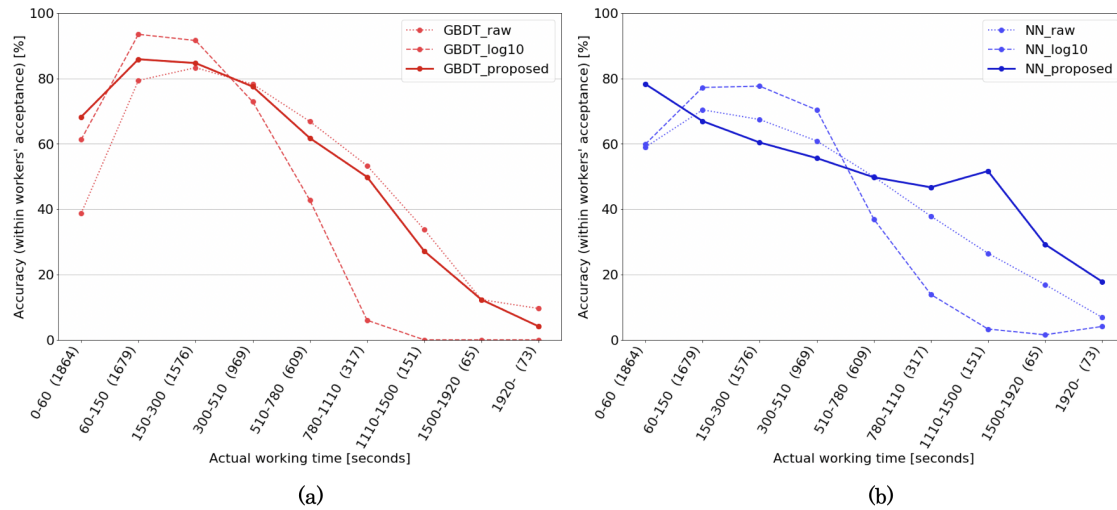


Figure 4-5: System performance comparison by working time categories, a) for GBDT and b) for a neural network. In the parenthesis after each actual working time category, the number of tested microtask data whose actual working time is within the range is shown. The working time categories are split based on the evaluation function for positive residual errors, but the accuracy includes both positive and negative errors.

model optimization is totally fair for workers; I had no intention of exploiting workers' feelings, but rather the results just showed that it was capable of predicting working times more accurately by solving the problem that the objective value-based optimization had.

However, we should also keep in mind objective meanings of time. The proposed evaluation function regards each working time prediction as “correct” if workers could tolerate its error regardless of its objective value. Therefore it should be further evaluated among the correct predictions on how much objective error they actually contained. In this chapter, supporting workers' decision making — by considering lost time caused by the actual amount of time difference — is the problem of risk management strategy design for workers and task scheduling, which could be another research topic and is out of the scope of this study.

Applicability of workers' tolerance to the real-world usage: In the proposed approach for defining CrowdSense, I posted microtasks that asked workers to analyze their daily crowd-work environment and express how they felt about prediction errors pertaining to simulated microtasks. Although my assumption regarding the simulated environment was sufficient for determin-

ing worker perception, it cannot be guaranteed that results obtained by using this approach would exactly match with worker opinions in real-world scenarios. For example, values of parameters, such as α , β , γ , and K , in Equation 4.1 can be different, or worker perception can possibly be more or less diverse compared to results obtained in this study. Thus, further investigation needs to be performed to determine these differences. However, collection of real-world data is very difficult owing to following reasons: *i*) there is no control on the experimenter side with regard to which pairs of predicted/actual working times must be used for questioning workers; this makes it difficult to sample enough data for each pair; *ii*) workers need to use to a certain kind of working-time prediction system to check predicted working times and complete a microtask to record the actual working time; this is simply too much work to be done for collection of a single sample; *iii*) there are other factors, such as requester preferences or microtask content, that add noise to or bias data, thereby making CrowdSense less generalized. Because of these reasons, a more detailed study design is necessary to collect real-world data for CrowdSense execution.

Pursuing a better prediction accuracy: The highest prediction accuracy observed in this study equaled $\sim 73\%$. I do not conclude this as the “real” upper limit of performance of the proposed working-time prediction approach, and suggest the following means to enhance the said performance. *i*) further feature engineering would contribute to attainment of higher accuracies. In addition to input features used in this study, more meaningful data, such as media content and dynamic elements rendered using JavaScript, can be extracted from microtasks. Other techniques, such as microtask category classification, based on natural language processing can also be employed; *ii*) As mentioned in Section 3.3, collected data demonstrates a long-tail distribution of working-time labels wherein most microtasks demonstrated short working times. Such bias in a dataset causes the trained prediction model to be over-optimized for microtasks with shorter working times. This issue can be addressed via collection of a larger dataset, which specifically aims at obtaining microtasks with longer working times by setting a higher bonus for workers accepting longer microtasks.

4.7 Conclusion

This study presented an approach to predict microtask working times based on CrowdSense — a technique to measure worker perception towards working-time prediction errors for model optimization and evaluation. The motivation behind this study was to help crowd workers estimate how long it would take them to complete a given microtask, thereby facilitating their search for more lucrative microtasks.

The proposed method first addressed the difficulty encountered in defining and gauging the “working time” and collecting associated microtask submission data. Next, this chapter presented the CrowdSense approach to quantify the subjective perception of workers towards errors in working-time prediction. This facilitates both optimization and evaluation of the model in terms of how workers perceive prediction results, which was previously not possible owing to exclusive use of objective values of the working time in seconds. Experimental results obtained in this study demonstrate that the proposed working-time prediction system was capable of predicting microtask working times with due worker acceptance in $\sim 73\%$ of all tested cases. The results also revealed that use of the CrowdSense-based model optimization enhances the prediction accuracy across a broad range of working times, which was not possible by using extant baseline approaches.

5

Conclusions

5.1 Summary of the Dissertation

In this dissertation, I proposed a machine learning-based system for predicting working times of microtasks that could be applied to any microtask that existed in a platform. Initially, I conducted a survey for worker strategies through tool and community usage to ensure the importance of the proposed method, followed by discussions on data collection method and feature engineering, which were key elements in building the predictive model. I also pointed out the inability of discussing how meaningful each prediction result (with a certain error) would be for the worker. I thus designed a method for quantifying workers' perception to prediction errors, given a relationship between a scale of and a distance between predicted working time and actual working time. By having such measurement adapt to the model design, it was expected that the predictive model would be optimized so that more prediction results are helpful for workers, as well as to be

evaluated based on it.

In **Chapter 2**, a quantitative analysis on crowd work environment through the usage of third-party tool and online community was conducted. To emphasize the importance of working time prediction, I conducted a survey among 360 AMT workers wherein I inquired about their working strategies. Having defined worker categories of the top 10% worker earnings as well as the top 50% and the bottom 50%, our survey results revealed that workers used tools and communities that helped them estimate “hourly wage” of microtasks more frequently if their worker earning was higher, which indicated that workers implicitly cared about the working time of microtasks.

In **Chapter 3**, I presented TurkScanner, a system for predicting working times of microtasks for any type of microtask existing in a crowd platform. The model accepted microtask-relevant information that was available to workers in platforms as input and returned working time in seconds as output. Specifically, I addressed three challenges in this work. The first challenge was related to defining a method for gauging working time. In order to cover all the expected worker behaviors while they are working on microtasks, I prepared two automatic methods and two manual methods (with the help of workers) for the purpose of recording working time candidates. The second challenge was concerned with collecting data used for model training and evaluation. To address the challenge, I developed a Chrome extension to be installed by AMT workers which collected microtask-relevant information from microtasks they visited as well as labels based on the recording methods. As a result of the data collection, I ended up hiring 83 workers and collecting 7303 microtask data records. The last challenge was related to designing and evaluating the predictive model. As feature engineering, I extracted 101-dimensional features based on microtask information (*e.g.*, microtask metadata and HTML elements), worker information (*e.g.*, working history and profiles), and requester information (*e.g.*, reputation data obtained from a third-party website.) Considering the dataset size and explainability, Gradient Boosting Decision Tree was selected for the predictive model. By visualizing a confusion matrix of the cross-validated evaluation results in heat maps, I was able to conclude that the predictive model was capable of roughly estimating the right scales of working times (estimating short microtasks as short and long microtasks as long).

In **Chapter 4**, I presented CrowdSense, a method for quantifying workers’ perception to errors of working time prediction. I first explained how I conducted a worker survey among AMT work-

ers that was aimed at formulating a relationship between microtask working time and its maximum acceptable prediction error. The survey assumed that participants hypothetically installed a worker tool that predicted working times of microtasks and suggested a pair of predicted value and actual value for the working time of a microtask. Participants answered whether the suggested error was acceptable for them or not. After gathering ~ 100 answers for each of the 918 pairs, I derived evaluation functions by fitting curves to the borders of the answers for the acceptance. Objective functions were also derived by integrating the reciprocals of the evaluation functions, which led to the formulation of a relationship between working time and workers' perception of working time. I then re-trained the predictive model to optimize it based on the obtained objective functions, and evaluated it based on the obtained evaluation functions. The evaluation results revealed that the re-trained model was capable of: *i*) predicting working times of 73% of all the tested microtasks within the error acceptance of workers, and *ii*) accurately predicting working times across more diverse scales of time.

5.2 Summary of Contributions

In this section, I summarized the contributions of my work in this dissertation from several different degrees of perspectives. Firstly, I will discuss its contribution in the context of assisting workers in crowd market, focusing on the newly-installed function in comparison to previous researches and tools. Next, I will expand the viewpoint to the whole crowd work out of the worker assistance, where I will explain its influences on requesters and the new usage of crowdsourcing as a technology. Lastly, I will describe the contribution as a new approach for building an artificial intelligence system.

Worker Assistance Perspective

With regard to assisting workers to select more lucrative microtasks, my work has two contributions. As mentioned since Chapter 1, the proposed method is capable of predicting working times of “any” microtasks that exists in the market. The current methods for calculating hourly wage, provided by Turkopticon and TurkerView, make such information available only for microtasks

that are already seen by other workers. This means that the suggested microtasks are more likely to be already popular and thus competitive among many workers, but other potentially-lucrative microtasks remain hidden in the market. In contrast, my work would be able to immediately suggest estimated working time or hourly price of every microtask, which means that workers would get the chance to seek more lucrative microtasks. This capability can be employed in enhancing the functionality of the market, such as in sorting microtasks by working time or hourly wage, or even filtering out non-lucrative microtasks from the microtask listings.

At the same time, to the best of my knowledge, my proposed system for predicting working time employed the first machine learning-based approach in the context of worker assistance. Before this study was carried out, we were not aware of whether such data-driven approach was possible in predicting working times, or if it was, what types of features affected its estimation. The evaluation results for the model performance had revealed that automatic working time prediction was possible based on microtask information. This indicates an interesting relationship between humans and machine learning — the machine learning model trained with data annotated by humans contribute for human performance to find microtasks that would contribute other types of machine learning model.

Whole Crowd Work Perspective

While my work was presented in this dissertation in the context of helping workers, I believe it has more contributions beyond those related to workers.

The working time prediction method would also be useful for requesters in setting prices for their microtasks. Similar to the difficulty for workers in estimating working time of posted microtasks, there are likewise challenges for requesters in setting the appropriate amounts of reward for their microtasks. Although a minimum wage is set by researchers (\$7.25/h), it is not yet widely known by requesters. Even for those who are knowledgeable on the standard, they would not know how long their microtasks would take to finish. Considering that the requesters themselves are those who know the most with regard to what is asked in the microtasks, they would likely underestimate the time required to complete them, thereby making the microtasks less lucrative than they think, which is a perceived problem that would increase unfairly-paid microtasks. To ad-

dress this problem, working time prediction method could be diverted into a requester helper tool. For instance, once a microtask is created by a requester, its information (meta data and HTML element information) along with reputation of the requester and profiles of an average worker could be input into the tool to get the estimated value of its working time. By using the estimated working time, the system could even calculate hourly wage and help requesters compare it with the minimum standard. Such system is expected to be lightweight and easy enough to be used by requesters through their general process of creating and posting microtasks. Furthermore, the more users of the system exist, the more training data of the model for the working time prediction can be collected, which makes the system even more generalized and stable for many requesters.

Knowing how long each microtask would take to finish would even be useful for predicting response time and/or ensuring QoS of crowd-powered systems. In crowd-powered systems wherein crowdsourcing is utilized as functions by being executed via APIs computer programs, response time of the systems is an important factor that determines system performance. The response time is determined by a sum of recruiting time and processing time. While methods for shortening recruiting time have been proposed by some research works (Bigham et al., 2010; Bernstein et al., 2012; Huang and Bigham, 2017), researches on controlling processing time are very few in numbers; there is one work for shortening the processing time to milliseconds, but it cannot be used in general purposes (Lundgard et al., 2018). In contrast, my work would enable to predict how long each response time from crowd would take to be returned with estimated working time. If response time of every microtask could be predicted before it is executed, crowd-powered systems can then ensure its QoS like those proposed in prior work (Khazankin et al., 2011, 2012) or even solve real-time scheduling problem in crowdsourcing as done in real-time operating systems (Buttazzo, 2011).

Artificial Intelligence Perspective

The methodology I presented in Chapter 5 for designing objective functions and evaluation functions based on quantified worker perception, oriented toward working time prediction error, suggests new ways in building an artificial intelligence framework that properly cooperates with humans. Nowadays, more and more artificial intelligence systems are developed to not only for full

automation of processes but also for assisting humans' decisions in various situations (e.g. business scenes, factories, farms, etc.). For models of such systems, it is important to try to make system outputs to be as helpful as possible for humans so that they can make better decisions; in my work, the system was optimized to maximize the likelihood of estimating working time to be accepted by workers. Such approach could be adapted to any type of system that uses regression to estimate objective values whose human perceptions are not yet determined. For example, inventory management could benefit from the same strategy, such that managers are assisted by a system to make decisions on how many units of each product should be ordered so that its stock would not be empty in future. By asking humans in charge of ordering the products about prediction errors they can accept for the decision, the system would be able to give better suggestions.

5.3 Limitations and Future Directions

Here, I will conclude this dissertation by mentioning the limitations and possible future directions of further studies.

5.3.1 Limitations

First, the data collection script I developed and explained in Chapter 4 is unable to extract some features due to technical difficulties. The script is only capable of scraping static contents in HTML; it is very hard to know what would be rendered dynamically by JavaScript in the micro-task, or what would workers be asked to do when the microtask navigates workers to an external website(s) where the real task resides. In fact, there are fairly a large number of microtasks in crowd markets that contain additional contents. Additional contents would make significant differences to working times. For instance, dynamic contents rendered by JavaScript would show tens or hundreds of transcription tasks in a row in the same website as they answer each task, or external URLs would navigate to websites where they are asked to answer tens or hundreds of survey questions, to register an account and log in, or to play games. However, there is no specific rule on the dynamic contents that appear and disappear in the interface, making it impossible to write a script to automate the data scraping. Moreover, asynchronous requests to external websites

by the script are often not allowed due to cross-domain policy of the world wide web.

Related to above, some of the input features relevant to media (*e.g.*, image, audio, video) do not take into account its content information. Every media that appears in a microtask may have different roles; for instance, an image may function as a target for transcription / bounding box drawing / object detection, an example image for the task, a background image, etc. Such role difference would also affect working time significantly. However, the selected features mentioned in Chapter 4 only include the numbers of those elements — other information about the media like its image size, audio duration, and video resolution, would be indicative features to be taken full advantage of in terms of working time estimation.

Next, my study was unable to conduct thorough validation of working time labels chosen by the participants from the four suggested options during data collection. This means that there is no guarantee if all of the selected options were the correct choices, and if all of them contained working times accurately. It is pretty hard to fully guarantee their data quality, because only workers know how long they actually spend on their microtasks.

5.3.2 Future Directions

This subsection discusses the possible future directions of the research presented in this dissertation corresponding to some applications proposed in the introduction. This dissertation focused on developing a fundamental technique for microtask working time prediction, thereby building a premise of implementing various applications; additional efforts are necessary to argue how the working time prediction technique should be used and improved in such applications. In below, considering the dissertation title “fair-trade crowdsourcing”, I will discuss application ideas and their possible challenges for *i*) worker assistance and *ii*) requester assistance.

Worker Assistance

The most direct and urgent application of my working time prediction technique is its application as a worker tool. As mentioned before, low worker wage is considered to be caused by a limitation of microtask information provided to workers so that they can easily locate profitable microtasks from a list of posted microtasks. Although existing worker tools can provide statistical

CHAPTER 5. CONCLUSIONS

The screenshot shows the Amazon MTurk Worker interface. At the top, there is a navigation bar with 'amazon mturk Worker', 'HITs', 'Dashboard', and 'Qualifications'. A search bar contains 'Search All' and a 'Filter' button. Below the navigation bar, the page title is 'All HITs Your HITs Queue'. The main content area is titled 'HIT Groups (1-20 of 79)' and includes 'Show Details' and 'Hide Details' buttons, and a dropdown for 'Items Per Page: 20'. The main table has columns: Requester, Title, HITs, Reward, Created, and Actions. A red dashed box highlights the 'HITs' and 'Reward' columns for the first 10 rows.

Requester	Title	HITs	Reward	Created	Actions
James Billings	Market Research Survey	33 s, \$1.08/hr	6,865 \$0.01	7m ago	Preview Accept & Work
Gaze Following	Where are they looking?	141 s, \$1.27/hr	1,134 \$0.05	17h ago	Preview Accept & Work
Scott Douglas Clary	Collect key contact from psycholo	106 s, \$1.01/hr	714 \$0.03	2d ago	Preview Accept & Work
purnawirman purnawirman	Help a child learn english!	83 s, \$0.43/hr	190 \$0.01	2d ago	Preview Accept & Work
Dick Co	Collect Data from a Website	143 s, \$0.25/hr	132 \$0.01	1d ago	Preview Accept & Work
Parmida Ghahremani	Nuclei Segmentation	148 s, \$6.07/hr	45 \$0.25	1d ago	Preview Accept & Work
Olaleye Olaitan	Label the requested items in the ir	116 s, \$0.92/hr	35 \$0.03	17h ago	Preview Accept & Work
samya Barkaoui	Data collection from website	110 s, \$0.65/hr	20 \$0.02	18h ago	Preview Accept & Work
samya Barkaoui	Data collection from website	121 s, \$0.30/hr	20 \$0.01	21h ago	Preview Accept & Work
Eugene Pak	Find Email Addresses	185 s, \$3.88/hr	19 \$0.20	9d ago	Preview Accept & Work
20bn	Record 16 short videos of opening	452 s, \$7.96/hr	16 \$1.00	1h ago	Preview Accept & Work

Figure 5-1: A sample interface for a worker tool that predicts microtask working time based on the proposed technique. The information shown in the dotted box is working time and/or hourly wage of each posted microtask additionally rendered by the tool. The system is capable of providing the information for every microtask regardless of whether or not it was previously completed by other workers.

working time estimates of microtasks based on working times spent by other workers who already completed them, they do not always work for all microtasks, considering that there are several microtasks newly posted in platforms, and thus not yet completed by any worker. By implementing the proposed technique into a new type of worker tool such as a browser extension or userscript, working times and hourly wages of every microtask in the list could be calculated and visualized in a worker interface (see Figure 5-1 for a worker tool example).

One of the possible relevant discussions is one concerned with a challenge of the usability design of the application in achieving the goals of workers for earning more wage. Taking the tool interface in Figure 5-1 for example, what kind of estimates should be provided and visualized to help workers — working time, hourly wage, or both of them? Working time can represent the amount of effort required to complete a microtask while it does not tell its profitability; on the other hand, hourly wage can represent profitability whereas it does not mean much for short-duration

microtasks (*e.g.*, is \$20/hr for a 10-second microtask good or bad?) Their cons could be covered by visualizing both working time and hourly wage, but excessive amounts of information would confuse the decision making of workers. Furthermore, discussions are also needed with regard to how such assistance would affect the achievement of the workers of their wage goals and what technique is additionally needed for it. To this end, user studies would be necessary by actually asking workers to use the developed worker tool.

Requester Assistance

To shape fair crowd markets, it would also be effective to build a framework for assisting requesters to set appropriate prices for microtasks they created. Such requester assistance is sometimes considered to be even more important than worker assistance (Singer and Mittal, 2011; Mason and Watts, 2009; Ikeda and Bernstein, 2016), in a sense that it would be a radical solution to the problem that prevents several workers from mistakenly picking cheaply-priced microtasks. The reasons that several requesters post such cheap microtasks are known to be that they do not really know how much the reward for their microtask should be, aside from that they are malicious requesters who intentionally set cheap prices. To support requesters, developing a web tool where they can upload their microtasks and where their proper prices can be suggested based on their estimated working times is essential.

Possible research questions here are: what type of interfaces and interactions can promote requesters design better microtasks, and how workers' decision making on their microtask design are affected by the tool. A low capability of microtask price setting can also be considered as a low capability of microtask designing. Such requesters needs to be guided for a technique for good microtask design, as also suggested in previous research (Faradani et al., 2011; Jain et al., 2017). To better design microtasks, requesters must be aware of the elements in their microtasks which necessitate more working time, and those that could be improved to make the jobs of workers more effective. However, discussing such features is not easy and is beyond the scope of this dissertation; for example, the tool could visualize contribution rate to working time estimation of microtask elements based on feature importance and attentions, or suggest the next action to improve the microtasks based on statistical analyses on interfaces of "good" microtasks that were

CHAPTER 5. CONCLUSIONS

previously posted in platforms. Such requester assistance tool design should also be developed by iterating implementation and user study by recruiting real requesters.

Bibliography

- Maha Alsayasneh, Sihem Amer-Yahia, Eric Gaussier, Vincent Leroy, Julien Pilourdault, Ria Mae Borromeo, Motomichi Toyama, and Jean-Michel Renders. Personalized and diverse task composition in crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 30(1): 128–141, 2017.
- Daniel W Barowy, Emery D Berger, Daniel G Goldstein, and Siddharth Suri. Voxpl: Programming with the wisdom of the crowd. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2347–2358. ACM, 2017.
- Benjamin B Bederson and Alexander J Quinn. Web workers unite! addressing challenges of online laborers. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 97–106. ACM, 2011.
- Janine Berg. Income security in the on-demand economy: Findings and policy lessons from a survey of crowdworkers. *Comp. Lab. L. & Pol'y J.*, 37:543, 2015.
- Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 33–42, 2011.
- Michael S Bernstein, David R Karger, Robert C Miller, and Joel Brandt. Analytic methods for optimizing realtime crowdsourcing. *arXiv preprint arXiv:1204.2995*, 2012.
- Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samuel White, et al. Vizwiz: nearly real-

BIBLIOGRAPHY

- time answers to visual questions. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology*, pages 333–342. ACM, 2010.
- Robin Brewer, Meredith Ringel Morris, and Anne Marie Piper. Why would anybody do this?: Understanding older adults’ motivations and challenges in crowd work. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 2246–2257. ACM, 2016.
- Steuart Henderson Britt. How weber’s law can be applied to marketing. *Business Horizons*, 18 (1):21–29, 1975.
- Susan L Bryant, Andrea Forte, and Amy Bruckman. Becoming wikipedian: transformation of participation in a collaborative online encyclopedia. In *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, pages 1–10, 2005.
- Giorgio C Buttazzo. *Hard real-time computing systems: predictable scheduling algorithms and applications*, volume 24. Springer Science & Business Media, 2011.
- Chris Callison-Burch. Crowd-workers: Aggregating information across turkers to help them find higher paying work. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.
- Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.
- Justin Cheng, Jaime Teevan, Shamsi T Iqbal, and Michael S Bernstein. Break it down: A comparison of macro-and microtasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 4061–4064. ACM, 2015.
- Chun-Wei Chiang, Anna Kasunic, and Saiph Savage. Crowd coach: Peer coaching for crowd workers’ skill growth. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW): 37, 2018.
- Lydia B Chilton, John J Horton, Robert C Miller, and Shiri Azenkot. Task search in a human

BIBLIOGRAPHY

- computation market. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 1–9. ACM, 2010.
- Derrick Coetzee, Seongtaek Lim, Armando Fox, Bjorn Hartmann, and Marti A Hearst. Structuring interactions for large-scale synchronous peer learning. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 1139–1152. ACM, 2015.
- Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.
- Mira Dontcheva, Robert R Morris, Joel R Brandt, and Elizabeth M Gerber. Combining crowdsourcing and learning to improve engagement and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3379–3388. ACM, 2014.
- David Durward, Ivo Blohm, and Jan Marco Leimeister. Crowd work. *Business & Information Systems Engineering*, 58(4):281–286, 2016.
- Siamak Faradani, Björn Hartmann, and Panagiotis G Ipeirotis. What’s the right price? pricing tasks for finishing on time. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.
- Gustav Theodor Fechner, Davis H Howes, and Edwin Garrigues Boring. *Elements of psychophysics*, volume 1. Holt, Rinehart and Winston New York, 1966.
- Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
- Snehalkumar Gaikwad, Nalin Chhibber, Vibhor Sehgal, Alipta Ballav, Catherine Mullings, Ahmed Nasser, Angela Richmond-Fuller, Aaron Gilbee, Dilrukshi Gamage, Mark Whiting, et al. Prototype tasks: improving crowdsourcing results through rapid, iterative task design. *arXiv preprint arXiv:1707.05645*, 2017.

BIBLIOGRAPHY

- Mary L Gray and Siddharth Suri. *Ghost Work: How to Stop Silicon Valley from Building a New Global Underclass*. Eamon Dolan Books, 2019.
- Daniel Haas and Michael J Franklin. Cioppino: Multi-tenant crowd management. In *Fifth AAAI Conference on Human Computation and Crowdsourcing*, 2017.
- Benjamin V Hanrahan, Jutta K Willamowski, Saiganesh Swaminathan, and David B Martin. Turk-bench: Rendering the market for turkers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1613–1616. ACM, 2015.
- Kotaro Hara and Jeffrey P Bigham. Introducing people with asd to crowd work. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 42–51. ACM, 2017.
- Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P Bigham. A data-driven analysis of workers’ earnings on amazon mechanical turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 449. ACM, 2018.
- Seth D Harris and Alan B Krueger. *A Proposal for Modernizing Labor Laws for Twenty-First-Century Work: The “Independent Worker”*. Hamilton Project, Brookings Washington, DC, 2015.
- Jeffrey Heer and Michael Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 203–212, 2010.
- Pamela J Hinds. The curse of expertise: The effects of expertise and debiasing methods on prediction of novice performance. *Journal of Experimental Psychology: Applied*, 5(2):205, 1999.
- Paul Hitlin. *Research in the Crowdsourcing Age, a Case Study: How Scholars, Companies and Workers are Using Mechanical Turk, a “gig Economy” Platform, for Tasks Computers Can’t Handle*. Pew Research Center, 2016.

BIBLIOGRAPHY

- John J Horton. The condition of the turking class: Are online employers fair and honest? *Economics Letters*, 111(1):10–12, 2011.
- John Joseph Horton and Lydia B Chilton. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 209–218. ACM, 2010.
- Jeff Howe. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.
- Ting-Hao K Huang and Jeffrey P Bigham. A 10-month-long deployment study of on-demand recruiting for low-latency crowdsourcing. In *In Proceedings of The fifth AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2017)*. AAAI, AAAI, 2017.
- Ting-Hao Kenneth Huang, Joseph Chee Chang, Saiganesh Swaminathan, and Jeffrey P Bigham. Evorus: A crowd-powered conversational assistant that automates itself over time. In *Adjunct Publication of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 155–157, 2017.
- Kazushi Ikeda and Michael S Bernstein. Pay it backward: Per-task payments on crowdsourcing platforms reduce productivity. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4111–4121. ACM, 2016.
- International Labour Office (ILO). Non-standard employment around the world: Understanding challenges, shaping prospects, 2016.
- Panagiotis G Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, Forthcoming, 2010a.
- Panagiotis G Ipeirotis. Mechanical turk, low wages, and the market for lemons. *A Computer Scientist in a Business School*, 27, 2010b.
- Lilly C Irani and M Silberman. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 611–620. ACM, 2013.

BIBLIOGRAPHY

- Lilly C Irani and M Silberman. Stories we tell about labor: Turkopticon and the trouble with design. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 4573–4586. ACM, 2016.
- Ayush Jain, Akash Das Sarma, Aditya Parameswaran, and Jennifer Widom. Understanding workers, developing effective tasks, and enhancing marketplace dynamics: a study of a large crowdsourcing marketplace. *Proceedings of the VLDB Endowment*, 10(7):829–840, 2017.
- Toni Kaplan, Susumu Saito, Kotaro Hara, and Jeffrey P Bigham. Striving to earn more: a survey of work strategies and tool use among crowd workers. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*, 2018.
- David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.
- Miranda Katz. Amazon mechanical turk workers have had enough, 2017. URL <https://www.wired.com/story/amazons-turker-crowd-has-had-enough/>.
- Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei hen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.
- Finn Kensing and Jeanette Blomberg. Participatory design: Issues and concerns. *Computer supported cooperative work (CSCW)*, 7(3-4):167–185, 1998.
- Nikhil Ketkar. Introduction to pytorch. In *Deep learning with python*, pages 195–208. Springer, 2017.
- Roman Khazankin, Harald Psaiar, Daniel Schall, and Schahram Dustdar. Qos-based task scheduling in crowdsourcing environments. In *International Conference on Service-Oriented Computing*, pages 297–311. Springer, 2011.
- Roman Khazankin, Daniel Schall, and Schahram Dustdar. Predicting qos in scheduled crowdsourcing. In *International Conference on Advanced Information Systems Engineering*, pages 460–472. Springer, 2012.

BIBLIOGRAPHY

- Joy Kim and Andres Monroy-Hernandez. Storia: Summarizing social media content based on narrative theory using crowdsourcing. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, pages 1018–1027, 2016.
- 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, 2008.
- Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 1301–1318. ACM, 2013.
- Yuki Koyama, Issei Sato, Daisuke Sakamoto, and Takeo Igarashi. Sequential line search for efficient visual design optimization by crowds. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.
- Siou Chew Kuek, Cecilia Paradi-Guilford, Toks Fayomi, Saori Imaizumi, Panos Ipeirotis, Patricia Pina, and Manpreet Singh. The global opportunity in online outsourcing. 2015.
- Tsuyoshi Kuroda and Emi Hasuo. The very first step to start psychophysical experiments. *Acoustical Science and Technology*, 35(1):1–9, 2014.
- Gierad Laput, Walter S Lasecki, Jason Wiese, Robert Xiao, Jeffrey P Bigham, and Chris Harrison. Zensors: Adaptive, rapidly deployable, human-intelligent sensor feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1935–1944, 2015.
- Leib Litman, Jonathan Robinson, and Tzvi Abberbock. Turkprime. com: A versatile crowdsourcing data acquisition platform for the behavioral sciences. *Behavior research methods*, 49(2): 433–442, 2017.

BIBLIOGRAPHY

- Fan Liu, Ajit Narayanan, and Quan Bai. Real-time systems. 2000.
- Alan Lundgard, Yiwei Yang, Maya L Foster, and Walter S Lasecki. Bolt: Instantaneous crowdsourcing via just-in-time training. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 467. ACM, 2018.
- David Martin, Benjamin V Hanrahan, Jacki O’Neill, and Neha Gupta. Being a turker. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 224–235. ACM, 2014.
- Winter Mason and Siddharth Suri. Conducting behavioral research on amazon’s mechanical turk. *Behavior research methods*, 44(1):1–23, 2012.
- Winter Mason and Duncan J Watts. Financial incentives and the performance of crowds. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 77–85. ACM, 2009.
- Brian McInnis, Dan Cosley, Chaebong Nam, and Gilly Leshed. Taking a hit: Designing around rejection, mistrust, risk, and workers’ experiences in amazon mechanical turk. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 2271–2282. ACM, 2016.
- Thomas P Moran and RJ Anderson. The workaday world as a paradigm for cscw design. In *Proceedings of the 1990 ACM conference on Computer-supported cooperative work*, pages 381–393, 1990.
- Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
- Michael Nebeling, Alexandra To, Anhong Guo, Adrian A de Freitas, Jaime Teevan, Steven P Dow, and Jeffrey P Bigham. Wearwrite: Crowd-assisted writing from smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3834–3846, 2016.
- Donald A Norman and Stephen W Draper. *User centered system design: New perspectives on human-computer interaction*. CRC Press, 1986.

BIBLIOGRAPHY

- Jacki O’neill and David Martin. Relationship-based business process crowdsourcing? In *IFIP Conference on Human-Computer Interaction*, pages 429–446. Springer, 2013.
- World Health Organization. *World health statistics 2016: monitoring health for the SDGs sustainable development goals*. World Health Organization, 2016.
- Stefan Palan and Christian Schitter. Prolific. ac—a subject pool for online experiments. *Journal of Behavioral and Experimental Finance*, 17:22–27, 2018.
- Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419, 2010.
- M Jordan Raddick, Georgia Bracey, Pamela L Gay, Chris J Lintott, Carie Cardamone, Phil Murray, Kevin Schawinski, Alexander S Szalay, and Jan Vandenberg. Galaxy zoo: Motivations of citizen scientists. *arXiv preprint arXiv:1303.6886*, 2013.
- Peter Reichl, Sebastian Egger, Raimund Schatz, and Alessandro D’Alconzo. The logarithmic nature of qoe and the role of the weber-fechner law in qoe assessment. In *2010 IEEE International Conference on Communications*, pages 1–5. IEEE, 2010.
- Yvonne Rogers. Exploring obstacles: integrating cscw in evolving organisations. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 67–77, 1994.
- Joel Ross, Lilly Irani, M Silberman, Andrew Zaldivar, and Bill Tomlinson. Who are the crowd-workers?: shifting demographics in mechanical turk. In *CHI’10 extended abstracts on Human factors in computing systems*, pages 2863–2872. ACM, 2010.
- Jeffrey M Rzeszotarski and Aniket Kittur. Instrumenting the crowd: using implicit behavioral measures to predict task performance. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 13–22. ACM, 2011.
- Susumu Saito, Tetsunori Kobayashi, and Teppei Nakano. Towards a framework for collaborative video surveillance system using crowdsourcing. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*, pages 393–396, 2016.

BIBLIOGRAPHY

Susumu Saito, Chun-Wei Chiang, Saiph Savage, Teppei Nakano, Tetsunori Kobayashi, and Jeffrey P Bigham. Turkscanner: Predicting the hourly wage of microtasks. In *The World Wide Web Conference*, pages 3187–3193. ACM, 2019.

Niloufar Salehi, Lilly C Irani, Michael S Bernstein, Ali Alkhatib, Eva Ogbe, Kristy Milland, et al. We are dynamo: Overcoming stalling and friction in collective action for crowd workers. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 1621–1630. ACM, 2015.

Gordon B Schmidt. Fifty days an mturk worker: The social and motivational context for amazon mechanical turk workers. *Industrial and Organizational Psychology*, 8(2):165–171, 2015.

Kjeld Schmidt and Liam Bannon. Taking cscw seriously. *Computer Supported Cooperative Work (CSCW)*, 1(1-2):7–40, 1992.

Boas Shamir and Ilan Salomon. Work-at-home and the quality of working life. *Academy of Management Review*, 10(3):455–464, 1985.

Varshita Sher, Karen G Bemis, Ilaria Liccardi, and Min Chen. An empirical study on the reliability of perceiving correlation indices using scatterplots. In *Computer Graphics Forum*, volume 36, pages 61–72. Wiley Online Library, 2017.

M Silberman, Joel Ross, Lilly Irani, and Bill Tomlinson. Sellers’ problems in human computation markets. In *Proceedings of the acm sigkdd workshop on human computation*, pages 18–21. ACM, 2010.

Yaron Singer and Manas Mittal. Pricing tasks in online labor markets. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

Brian L Sullivan, Christopher L Wood, Marshall J Iliff, Rick E Bonney, Daniel Fink, and Steve Kelling. ebird: A citizen-based bird observation network in the biological sciences. *Biological conservation*, 142(10):2282–2292, 2009.

BIBLIOGRAPHY

- John C Tang, Manuel Cebrian, Nicklaus A Giacobe, Hyun-Woo Kim, Taemie Kim, and Douglas” Beaker” Wickert. Reflecting on the darpa red balloon challenge. *Communications of the ACM*, 54(4):78–85, 2011.
- William Thies, Aishwarya Ratan, and James Davis. Paid crowdsourcing as a vehicle for global development. In *CHI Workshop on Crowdsourcing and Human Computation*, 2011.
- Melissa A Valentine, Daniela Retelny, Alexandra To, Negar Rahmati, Tulsee Doshi, and Michael S Bernstein. Flash organizations: Crowdsourcing complex work by structuring crowds as organizations. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, pages 3523–3537, 2017.
- RUI Wang. Two’s company, three’s a crowd: can h2s be the third endogenous gaseous transmitter? *The FASEB journal*, 16(13):1792–1798, 2002.
- Mark E Whiting, Grant Hugh, and Michael S Bernstein. Fair work: Crowd work minimum wage with one line of code. 2019.
- Robert Sessions Woodworth and Harold Schlosberg. *Experimental psychology*. Oxford and IBH Publishing, 1954.
- Meng-Han Wu and Alexander James Quinn. Confusing the crowd: Task instruction quality on amazon mechanical turk. In *Fifth AAAI Conference on Human Computation and Crowdsourcing*, 2017.
- Louis E Yelle. The learning curve: Historical review and comprehensive survey. *Decision sciences*, 10(2):302–328, 1979.

Research Achievements

JOURNAL PAPER PUBLICATION

- 2019** Susumu Saito, Chun-Wei Chiang, Saiph Savage, Teppei Nakano, Tetsunori Kobayashi, and Jeffrey P. Bigham, Predicting the Working Time of Microtasks Based On Workers' Perception To Prediction Errors, *Human Computation* 6.1 (2019): 192-219.

INTERNATIONAL CONFERENCE PAPER PUBLICATIONS

(PEER REVIEWED)

- 2019** Susumu Saito, Teppei Nakano, Tetsunori Kobayashi, and Jeffrey P. Bigham. MicroLapse: Measuring Workers' Leniency To Prediction Errors of Microtasks' Working Times. In Proceedings of the 22nd ACM Conference on Computer Supported Cooperative Work and Social Computing Companion (CSCW '19 Companion). ACM, New York, NY, USA, 1053-1056. DOI: <https://doi.org/10.1145/3311957.3359466>
- 2019** Ryosuke Hyodo, Saki Yasuda, Yusuke Okimoto, Susumu Saito, Teppei Nakano, Makoto Akanabe, Tetsunori Kobayashi, and Tetsuji Ogawa. Two-stage calving prediction system: Exploiting state-based information relevant to calving signs in Japanese black beef cows. ECPLF 2019. Cork, Ireland.
- 2019** Kazuma Sugawara, Susumu Saito, Teppei Nakano, Makoto Akabane, Tetsunori Kobayashi, and Tetsuji Ogawa. Calving prediction from video: Exploiting behavioral information relevant to calving signs in Japanese black beef cows. ECPLF 2019. Cork, Ireland.
- 2019** Susumu Saito, Chun-Wei Chiang, Saiph Savage, Teppei Nakano, Tetsunori Kobayashi, and Jeffrey P. Bigham. TurkScanner: Predicting the Hourly Wage of Microtasks. In The World Wide Web Conference (WWW '19). ACM, New York, NY, USA, 3187-3193. DOI: <https://doi.org/10.1145/3308558.3313716>

BIBLIOGRAPHY

- 2018** Toni Kaplan*, Susumu Saito*, Kotaro Hara, and Jeffrey P. Bigham. Striving to earn more: a survey of work strategies and tool use among crowd workers. In Sixth AAI Conference on Human Computation and Crowdsourcing. (* ... Equal contribution)
- 2016** Susumu Saito, Teppei Nakano, Makoto Akabane, and Tetsunori Kobayashi. Evaluation of Collaborative Video Surveillance Platform: Prototype Development of Abandoned Object Detection. In Proceedings of the 10th International Conference on Distributed Smart Camera (ICDSC '16). ACM, New York, NY, USA, 172-177. DOI: <https://doi.org/10.1145/2967413.2967416>
- 2016** Susumu Saito, Tetsunori Kobayashi, and Teppei Nakano. Towards a Framework for Collaborative Video Surveillance System Using Crowdsourcing. In Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion (CSCW '16 Companion). ACM, New York, NY, USA, 393-396. DOI: <https://doi.org/10.1145/2818052.2869074>

WORKSHOPS AND TALKS

- 2018** Susumu Saito. Designing For Real-Time Computing-Based Microtask Scheduling. AAI HCOMP2018 Doctoral Consortium. Zurich, Switzerland. July.
- 2018** Susumu Saito, Jeffrey P. Bigham, Teppei Nakano, Tetsunori Kobayashi. Toward A Crowdsourcing Platform For Real-Time Computing-Based Microtask Scheduling. Asian CHI Symposium: Emerging CHI Research Collection. Montréal, Canada. April.
- 2017** Susumu Saito. Activities in Graduate Program of Embodiment Informatics. +R Leading Seminar, Ritsumeikan University. Shiga, Japan. February.
- 2016** Kazuya Ueki, Kotaro Kikuchi, Susumu Saito, and Tetsunori Kobayashi. Waseda at TRECVID 2016: Ad-hoc Video Search. TRECVID 2016. Gaithersburg, MD. November.

BIBLIOGRAPHY

DOMESTIC CONFERENCE PAPERS

- 2019** 齋藤奨, 中野鐵兵, 小林哲則. TurkScanner: マイクロタスクの時給推定. 人工知能学会 第33回全国大会. 新潟. 6月.
- 2019** 沖本祐典, 齋藤奨, 中野鐵兵, 赤羽誠, 小林哲則, 小川哲司. 肉牛の分娩検知システムにおけるクラウドソーシングを用いた誤通報の抑制. 人工知能学会 第33回全国大会. 新潟. 6月.
- 2019** 兵頭亮介, 齋藤奨, 中野鐵兵, 赤羽誠, 小林哲則, 小川哲司. 画像から得られる牛の身体情報に基づく分娩予兆検知. 人工知能学会 第33回全国大会. 新潟. 6月.
- 2019** 沖本祐典, 齋藤奨, 中野鐵兵, 赤羽誠, 小林哲則, 小川哲司. 映像による肉牛分娩開始検知システムの早期運用のためのクラウドソーシングを用いた誤通報抑制. 電子情報通信学会 パターン認識・メディア理解研究会 (PRMU). 東京. 5月.
- 2018** 沖本祐典, 菅原一真, 齋藤奨, 中野鐵兵, 赤羽誠, 小林哲則, 小川哲司. 牛の分娩予兆として映像から観測可能な状態の検知. 人工知能学会 第32回全国大会. 鹿児島. 6月.
- 2016** 齋藤奨, 中野鐵兵, 小林哲則. クラウドソーシングを用いた協調型ビデオ監視システムフレームワークの提案. 第78回情報処理学会全国大会. 埼玉. 3月.
- 2016** 齋藤奨, 中野鐵兵, 小林哲則. クラウドソーシングを用いた協調型ビデオ監視システムのプロトタイプ設計. 2016年電子情報通信学会全国大会. 福岡. 3月.
- 2015** 齋藤奨, 小林哲則. 学習者の知識補完のためのオンライン講義コンテンツ連携フレームワーク. 情報処理学会 コンピュータと教育研究会132回研究発表会. 福井. 12月.

BIBLIOGRAPHY

PATENTS

2016 Condition Monitoring System (Japan 2016-165131)

FELLOWSHIPS

2019 JSPS Researcher Fellowship (DC2) (April 2019–March 2021)

2015 Graduate Program for Embodiment Informatics (April 2015–March 2019)