Turkish Judge: A Peer Evaluation Framework for Crowd Work Appeals

Edward Cohen, Mukund Venkateswaran, Nivedita Sankar, Chris Callison-Burch
University of Pennsylvania
3401 Walnut Street
Philadelphia, Pennsylvania 19104
{edcohen, mukundv, nsankar, ccb}@seas.upenn.edu

Abstract

We present our work in progress platform Turkish Judge, a crowd-driven adjudication system for rejected work on Amazon Mechanical Turk. The Mechanical Turk crowdsourcing platform allows Requesters to approve or reject assignments submitted by Workers. If the work is rejected, then Workers aren't paid, and their reputation suffers. Currently, there is no built-in mechanism for Workers to appeal rejections, other than contacting Requesters directly. The time it takes Requesters to review potentially incorrectly rejected tasks means that their costs are substantially higher than the payment amount that is in dispute. As a solution to this issue, we present an automated appeals system called Turkish Judge which employs crowd workers as judges to adjudicate whether work was fairly rejected when their peers initiate an appeal.

Introduction

Amazon Mechanical Turk (AMT) is a crowdsourcing platform developed by Amazon which allows businesses and individuals (Requesters) to hire Workers (sometimes called Turkers) to completed microtasks on the platform. Tasks uploaded by a Requester to the Amazon Mechanical Turk marketplace to be completed by Turkers are called HITs (for Human Intelligence Tasks). AMT allows Requesters to review HITs submitted by Workers. Requesters may approve the HITs, in which case the Worker is paid, or they may reject the HITs, in which case the Worker is not paid.

Beyond wasted time and not receiving compensation for their work, a Worker is additionally penalized since a rejection diminishes their rating in AMT's reputation system. Each Turker has an associated score based on their past approval rate. Requesters restrict their work to Workers whose approval rate is above a specified threshold. Whenever a Worker receives a rejection their approval rate decreases, which limits the work they are able to access on the platform.

Requesters are able to reject HITs submitted by Turkers for any reason whatsoever, and AMT provides no mechanism for Workers to appeal the rejection beyond emailing the Requester. This raises obvious concerns over fairness. Workers are aware of this issue and in a 2019 survey, 69% of Workers cited unfair rejections in their top three grievances with the platform (Whiting, Hugh, and Bernstein 2019). Turkers have means to label and make others aware of unfair Requesters through either TurkOpticon (Irani and Silberman 2013) or popular forums like Turker Nation. On TurkOpticon, Turkers can rate Requesters on their "fairness". This attribute is a 1 to 5 scale for how fair a Requester is in approving or rejecting work.

Some of the rejections on the platform are warranted and weed out malicious Workers. Mechanical Turk has built-in functionality allowing Requesters to overturn rejections, but Workers have little recourse for asking for this. Mechanical Turk supports Workers emailing Requesters directly asking them to overturn the rejection. However, Requesters rarely read or respond to these because of the time costs associated with a manual resolution of the grievance (Irani and Silberman 2013). Therefore, any proposed solution to this issue must be minimum effort to Requesters.

To address these issues, we built Turkish Judge, a platform to give Workers a way of appealing rejected work, and to simplify the Requester's job of adjudicating their appeals by allowing the Requester to hire other crowd workers to act as peer reviewers of the rejected work. The mechanism posts the HIT in question and an explanation provided by the appellant on the AMT marketplace for other Turkers to adjudicate on whether the rejection was fair or unfair. The buy-in from requesters is that they upload a file containing their HIT results and check in to see the status of their rejected HITs. This allows the Requester to overturn the rejection or uphold it with minimal effort. It reduces the burden placed on Requesters to manually review appeals, and provides for the fairer treatment of Workers.

Related Work

The literature on potential solutions is written in the context of design considerations for the platform beyond just unfair rejections. (Bederson and Quinn 2011; Martin et al. 2014). Generally, Turkers are identified as an invisible workforce with calls being made for more transparency and communication on the platform (Martin et al. 2014). Toward this end, services like TurkOpticon and Crowd Workers, as well as platforms like Turker Nation provide Workers with information into Requesters and the marketplace (Irani and Silberman 2013; Callison-Burch 2014). On the issue of rejections specifically, Bederson and Quinn suggest that Requesters should at least have to email Turkers with an explanation for why their work was rejected (Bederson and Quinn 2011). Turkers have suggested three options: they are given the chance to redo the work, the Requesters should not be allowed to keep rejected work, and similar to Bederson and Quinn, that Requesters should be obligated to elucidate a reason for the rejection (Felstiner 2011).

The externally-developed worker tool TurkerView released a rejection dispute tool, TurkerView Bridge, this past February (TurkerView 2020). Turkers can submit a text, image, or video reason for why the work should not have been rejected. This appeal gets sent to the Requester for them to review. TurkerView Bridge provides information for the requester on overturning rejections and a properly formatted csv for one to upload to reverse rejections. This solution reduces the friction in the rejection dispute process, however it still relies on Requesters to take the time to review disputed rejections.

Design Rationale

Currently, when a Turker submits a grievance request regarding their rejected work, the Requester must read the petition in full, manually search through the HIT batch data for the corresponding row entry, assess its quality, and return a final judgement. This process takes a nontrivial amount of time. Given that the median hourly wage for Turkers is around two dollars an hour (Hara et al. 2017), the Requester incurs more cost reviewing the petition than they would have paid originally. Irani captured this sentiment in a quote from a Requester who said "[a Requester] cannot spend time exchanging email. The time you spent looking at the email costs more than what you paid [the Worker]." (Irani 2015). This imbalance means that the petitioning Turker frequently receives no response, much less payment and a reputation increase.

As a result, a mechanism that removes the time burden associated with Requesters manually reviewing appeals is necessary. A mechanism which allows Requesters to adjudicate appeals quickly and at low cost could enable a marketplace that is both easy for Requesters and fairer to Workers. We take this into account when designing the mechanism workflows.

Turkish Judge

Turkish Judge is built as a simple Flask web application. Requesters can create an account to upload batches to the platform, and Workers can then simply enter a corresponding WorkerID and HITID in order to submit an appeal. We outline the workflows of a Requester, a Worker submitting an appeal, and a crowdworker acting as an Judge.

Requester Workflow

After a batch is completed and reviewed by a Requester, the Requester should upload the batch information to Turkish Judge for adjudication. Here, we have access to the relevant information necessary for the adjudication task including HITIDs, WorkerIDs, the HIT in question, and whether the task was accepted or rejected. This information is stored in our database to allow rejected Workers to appeal their HITs. Once a Turker submits an appeal and the task is adjudicated, the Turker and the Requester will both be notified so that the Requester can act correspondingly to overturn the rejection for the Turker if the rejection is deemed unfair.

This allows for Requesters to spend much less time dealing with messages from Turkers who were rejected and introduces an expert third party to analyze the completion of the HIT.

To overturn rejected HITs, Requesters have the option of uploading a csv file of Worker data with "Approve" and "Reject" columns. After a verdict is reached on all appealed HITs in a batch, the Requester will be able to download a csv file containing the verdict for each HIT. A Requester will simply be able to upload this csv file to Mechanical Turk in order to reverse the rejection on the overturned HITs. This greatly accelerates the Requester workflow from manual adjudication, as they simply have to upload one csv file rather than individually dealing with each Worker grievance.

Therefore the only work required from the Requester is to upload completed HITs, and overturn rejections if and when they are deemed unfair. Both of these take brief amounts of time and thus are minimum effort options to Requesters.

Worker Workflow

In order to appeal a rejected HIT, Workers provide the ID of the HIT in question along with their Mechanical Turk WorkerID in order to identify the corresponding HIT. The Turker will be prompted to enter an explanation for their completion of the HIT describing why the rejection should be overturned. Once a Worker submits an appeal, an adjudication task is created for 3-5 other Turkers to complete.

Adjudicator Workflow

An adjudication task is comprised of the original HIT in question, an optional appeal message from the Turker, the rejection message from the Requester, as well as some instructions on how to adjudicate the HIT as objectively as possible. The Judge who is given this task will be able to respond either that the rejection should be upheld or that it should be overturned. We aggregate the responses that we get from the Judges to make a decision on the task in question.

Analysis of Adjudication Quality (In Progress)

A prominent concern with such a mechanism is adjudication quality. More specifically, how can effortful and truthful adjudication be incentivized? One concern is that because Turkers are predisposed to feel Requesters rejections are unfair, perhaps Turkers would simply side with their peers when adjudicating HITs. This issue is of the utmost importance in the fruitfulness of our system, since there is no use in a corrupt judicial system.

To empirically estimate the quality of our adjudication mechanism, we are simulating batches of rejected work to analyze whether Judges' adjudications align with our expected outcomes. Each of the appeals has a ground truth label for whether it should be overturned or not. We believe that Turkers have the ability to effectively determine whether or not a rejected HIT was actually properly completed because they are experts in completing HITs. Using the gathered data, we seek to measure the productivity and behavior of Workers on the adjudication task.

We plan on extending this analysis to a variety of different types of tasks, but we show initial results on a simple text classification task below as proof of concept.

Results

Results of our adjudications tasks are summarized in the appendix [2, 3]. Note that each verdict is representative of five Workers, making each individual decision made by the judges more statistically significant than if adjudicated simply by one or three Workers.

Analysis

To quantify the quality of adjudication through the formulated crowd based appeals system, we focus on the precision and recall of overturning rejected HITs.

In determination of true labels for HIT completion, a generous threshold for properly completed HITs might be an accuracy of .75. Using this as the true labels for our empirical experimentation, we find that there are very few false positive predictions made by the Judges. This shows that Judges very rarely reverse rejections on improperly completed HITs.

On the other hand, we find quite a few false negative predictions. This shows that our Judges frequently upheld rejections when we would expect the rejection to be overturned according to our performance threshold. We break this down further in [Figure 3] to show the threshold of accuracy expected by Judges when overturning a rejection. With an accuracy even as high as .83, we find that Turkers still uphold rejection on the majority of the appeals. Only on HITs completed with perfect accuracy do we find that Judges overturn a majority of the HITs.

These results indicate Judges' proficiency in the analysis of HIT completion quality as we can see that the rejection reversal rate increases monotonically with the accuracy of HIT completion. Moreover, these results indicate that Judges uphold higher standards of HIT completion quality than we expected.

Conclusion

Ongoing work on this project includes a more thorough empirical evaluation that incorporates a variety of HITs, a study of Requester's incentives for a platform, surveys to estimate whether Turkish Judge changes Workers' perceptions of fairness of rejections or willingness to work with Requesters who use it, and updates to the interface and functionality of the website. The Requester dashboard, and Worker appeal page are shown in [Figure 4].

Acknowledgements

This research was sponsored by the National Science Foundation's Future of Work at the Human-Technology Frontier program under award number 1928474.

References

Bederson, B. B., and Quinn, A. J. 2011. Web workers unite! addressing challenges of online laborers. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*. 97–106.

Callison-Burch, C. 2014. Crowd-workers: Aggregating information across turkers to help them find higher paying work. In *The Second AAAI Conference on Human Computation and Crowdsourcing (HCOMP-2014)*.

Felstiner, A. 2011. Working the crowd: Employment and labor law in the crowdsourcing industry. *Berkeley J. Emp. & Lab. L.* 32:143.

Hara, K.; Adams, A.; Milland, K.; Savage, S.; Callison-Burch, C.; and Bigham, J. P. 2017. A data-driven analysis of workers' earnings on amazon mechanical turk. *CoRR* abs/1712.05796.

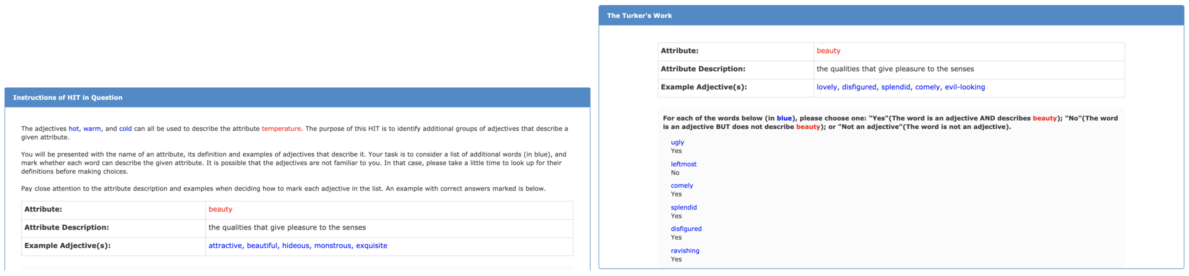
Irani, L. C., and Silberman, M. S. 2013. Turkopticon: Interrupting worker invisibility in Amazon Mechanical Turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 611–620.

Irani, L. 2015. The cultural work of microwork. *New Media & Society* 17(5):720–739.

Martin, D.; Hanrahan, B. V.; O'Neill, J.; and Gupta, N. 2014. Being a Turker. In *Proceedings of the 17th ACM conference on Computer Supported Cooperative Work & Social Computing*, 224–235.

TurkerView. 2020. Building bridges: TurkerView launches MTurk bridge — rejection disputes.

Whiting, M. E.; Hugh, G.; and Bernstein, M. S. 2019. Fair work: Crowd work minimum wage with one line of code. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7, 197–206.



Instructions of HIT in Question

The adjectives hot, warm, and cold can all be used to describe the attribute temperature. The purpose of this HIT is to identify additional groups of adjectives that describe a given attribute.

You will be presented with the name of an attribute, its definition and examples of adjectives that describe it. Your task is to consider a list of additional words (in blue), and mark whether each word can describe the given attribute. It is possible that the adjectives are not familiar to you. In that case, please take a little time to look up for their definitions before making choices.

Pay close attention to the attribute description and examples when deciding how to mark each adjective in the list. An example with correct answers marked is below.

Attribute:	beauty
Attribute Description:	the qualities that give pleasure to the senses
Example Adjective(s):	attractive, beautiful, hideous, monstrous, exquisite

The Turker's Work

Attribute:	beauty
Attribute Description:	the qualities that give pleasure to the senses
Example Adjective(s):	lovely, disfigured, splendid, comely, evil-looking

For each of the words below (in blue), please choose one: "Yes"(The word is an adjective AND describes beauty); "No"(The word is an adjective BUT does not describe beauty); or "Not an adjective"(The word is not an adjective).

ugly
Yes

leftmost
No

comely
Yes

splendid
Yes

disfigured
Yes

ravishing
Yes

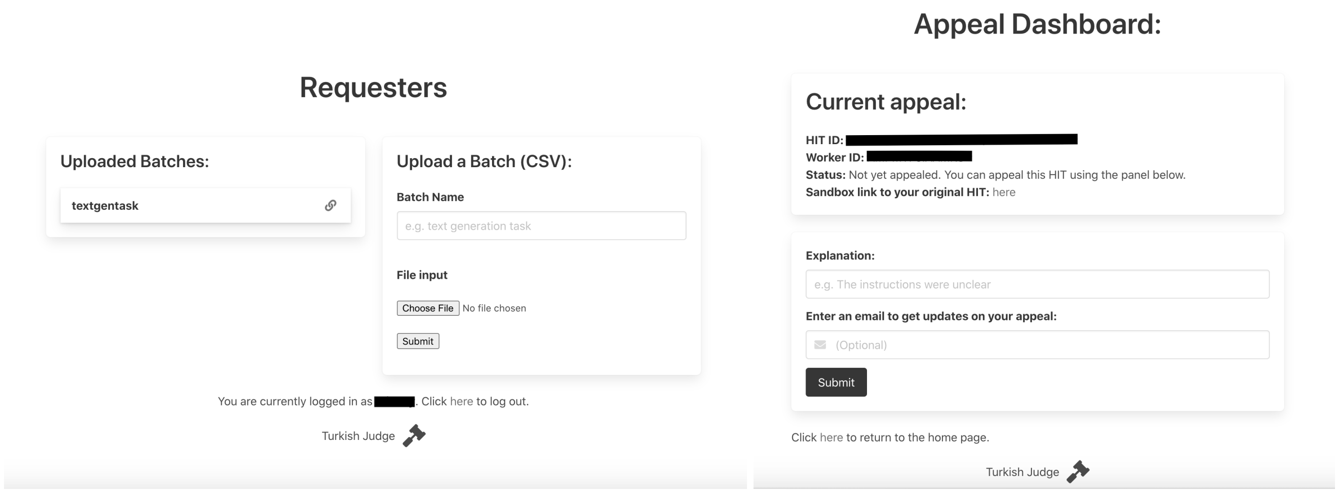
Figure 1: Depiction of original rejected HIT within the appeal task.

Actual / Verdict	**V: Overturn**	**V: Reject**
A: Overturn	$TP = 27$	$FN = 23$
A: Reject	$FP = 2$	$TN = 46$

Figure 2: Confusion matrix for HIT adjudication, based on a Requester hypothetically accepting all work with an accuracy threshold of 75%.

Accuracy:	0.0 - 0.50	.67	.83	1.0
% of HITs overturned:	2.5%	12.50%	30.77%	79.17%

Figure 3: Accuracy of our simulated rejected HIT vs. percentage of HITs overturned. Judges were more likely to overturn the rejections on HITs that had been completed with high accuracy than for HITs that were erroneous.



Requesters

Uploaded Batches:

textgentask

Upload a Batch (CSV):

Batch Name

e.g. text generation task

File input

Choose File No file chosen

Submit

You are currently logged in as [redacted]. Click here to log out.

Turkish Judge

Appeal Dashboard:

Current appeal:

HIT ID: [redacted]
Worker ID: [redacted]
Status: Not yet appealed. You can appeal this HIT using the panel below.
Sandbox link to your original HIT: here

Explanation:

e.g. The instructions were unclear

Enter an email to get updates on your appeal:

(Optional)

Submit

Click here to return to the home page.

Turkish Judge

Figure 4: The Requester dashboard and the page to submit an appeal.