STANFORD UNIVERSITY

coco lab

Mturk-tools

From coco lab

Here are some instructions on how to get started doing experiments on Mechanical Turk. The net result is that you will write your experiment as a webpage, which your subjects will see as a frame within the Mechanical Turk window. By all reports, Turkers like this better than having to go to an external site and enter a code, etc. Getting external HITs posted to CLT is fairly straightforward, except for the part where you actually have to write your experiment.

Please feel free to post edits to this, or email me (danlassiter@stanford.edu) if you think something here needs to be clarified.

This section contains the more technical stuff, and the next section contains practical advice about IRB, reimbursements, how to conduct good experiments, manage workers, etc.

1. Download Amazon's Command Line Tools (http://aws.amazon.com/developertools/694) (CLT). Open the file GetStarted.html included in the download. Follow the instructions to create a mTurk requester account, sign up for Amazon Web Services, and install the Command Line Tools (follow steps 2-3 below before testing that CLT is working). (If you have Java-related trouble with this step, read the * at the bottom of this page.) (if you have already made an account on sandbox but when you try to post a HIT there, you get "error #1 go to sandbox and complete registration", log out and then log back in, and then it will ask you additional registration questions.)

2. The identifiers that you are instructed to insert in the file mturk.properties in the /bin folder of your CLT installation are part of your MTurk requester information. To get them go to the AWS identifier page (http://s3.amazonaws.com/mturk/tools/pages/aws-access-identifiers/aws-identifier.html) after you've created a Requester account.

3. Also edit mturk.properties to change the service URL from "http://"... to "https://"... (As of June 8, 2012 it appears that Amazon is requiring you to use https instead of http, but the current version of CLT doesn't reflect this.)

4. Write your experiment in HTML and JavaScript. Include the legal blurb and lab logo on the first page and then have the detailed instructions on a separate page, as in Dan's animal (http://www.stanford.edu /~danlass/experiment/animals.html) and adjective/adverb experiment (http://www.stanford.edu/~danlass /experiment/adj-adv/index.html) .

Also, include some code that makes sure that Workers can't start the experiment without first accepting the HIT. Otherwise, they can go through the entire experiment and then discover that nothing happens because, as far as Amazon is concerned, they're not working for you. Here's an example from Dan that accomplishes this using Long's library, mmturkey: in the javascript file (http://www.stanford.edu /~danlass/experiment/adj-adv/experiment.js) for this experiment, the stepExperiment function access the turk.previewMode variable (provided by mmturkey) to detect whether the worker is currently only previewing the HIT.

5. To manage the interface with mTurk, use Long's library, mmturkey (http://github.com/longouyang /mmturkey) (see also the even-odd example (http://longouyang.github.com/even-odd/docs/even-odd.html) on how to use the library). This makes things much simpler; experimental results stay on the user's machine and get passed to Amazon at the end; Amazon will compile results for you and let you retrieve them. If you haven't used HTML and JavaScript before, use the even-odd experiment as a template and search Google for lots of detailed introductions and examples.

6. To actually post your experiment, download and unzip Dan's submiterator script (http://github.com /danlassiter/Submiterator/zipball/master) . Put the two included files in a dedicated folder for your experiment. Open the README file in a text editor and follow the instructions, and you should be up and running in no time. (Don't change the name of this file, and don't move any files once you've posted your experiment!) The only caveat is I haven't tested this on Windows so it may not (probably won't) work.

7. Note that webpages don't always display the same way in an mTurk frame that they do when you open them in a normal browser window. Use the sandbox setting in Submiterator to find out whether your experiment displays as expected before going live.

8. To get results, use Dan's submiterator script or go to the Manage tab and click the tiny link in the upper right hand corner that says Manage Hits Individually (https://requestersandbox.mturk.com/mturk /manageHITs) .

Before running experiments
You will need to do the CITI training for human-subjects experiments, then you will need to be added to the IRB protocol as an experimenter. For now, see Noah about this.

Reimbursements/advances
As a researcher, you have two options for remuneration - getting reimbursed after the fact, or applying

for an advance. Officially, for reimbursements you need to submit (1) a receipt and (2) a list of Amazon worker ID's, payments, and fees for everyone that did your HIT to Rani (or whoever your PI's administrative assistant is). I've actually had success getting reimbursed with just a printout of the emails from Amazon ("Your purchase of Mechanical Turk prepaid HITs succeeded!") and a copy of my credit card statements with the lines circled where the charges from Amazon are. But, I can't promise that that will work for everyone. Reimbursement is somewhat slow, but can be sped up somewhat if you also submit a direct deposit form for reimbursements.

For an advance, you need to submit a request and an agreement. You can request up to $1,000 for an advance for a period of up to six months - this is nice, because it ends up being less paperwork overall. A downside is that it takes a while longer (2 to 3 weeks) to get the advance. After the advance ends, you have a couple of weeks to submit all the receipts that you accumulated during that time. {todo: upload example request and agreement}.

Legal blurb: put this at the beginning of your HITs.

Legal information: By answering the following questions, you are participating in a study being performed by cognitive scientists in the Stanford Department of Psychology. If you have questions about this research, please contact YOURNAMEHERE at YOUREMAILHERE or Noah Goodman, at ngoodman@stanford.edu. You must be at least 18 years old to participate. Your participation in this research is voluntary. You may decline to answer any or all of the following questions. You may decline further participation, at any time, without adverse consequences. Your anonymity is assured; the researchers who have requested your participation will not receive any personal information about you.

Tactics and Analytical Hints (or, how I learned to stop worrying and love the Turk)

1. In the legalese, make your email address clickable but not Noah's. Some times people will email with issues - you want to make sure that the emails go directly to you, so that they're not bugging Noah and also so that the issues get resolved faster.

2. Sign up as a requester on Turkernation (http://turkernation.com/) and introduce yourself politely on the Requesters' introductions page, explaining briefly who you are and what you do, and respond quickly to any responses you get. Turkers love it and will be more inclined to do your HITs (and more carefully) when you engage with them and give them opportunities for feedback, since they mostly see requesters as evil faceless corporations out to exploit them. When I'm running an experiment I monitor Turkernation for mentions of my name so that I can respond quickly if there are any concerns or negative feedback. Also, try out the Turkopticon widget for Firefox. A lot of Turkers use this since it provides quick summary statistics about people's previous experience with a requester. If you get bad ratings on Turkopticon, you'll have a hard time getting people to do your HIT.

3. VERY IMPORTANT:

a. Thinking carefully about your workers' comfort and happiness is not just for their benefit, it's for yours as well. Turkers are well-organized and communicate with each other (in particular via Turkernation), and if you get a bad reputation you'll get bad data or no data at all. In particular, it's important that you're paying enough to make the task worth their time. If they think they're being exploited, you'll quickly find you can't run your experiments. (More on effective rates below)

b. Whichever email account you give to Amazon, monitor it whenever you have a HIT up. Workers will email you if they have difficulty or if your code is buggy, etc. If you respond quickly and politely you'll get a good Turkopticon rating for communication, and subsequent workers will be more likely to do your experiments.

c. If you're posting a batch of HITs, make sure that either (a) workers can do all of the HITs in the batch, or (b) you state VERY CLEARLY IN THE TITLE what restrictions there are. If you choose the latter course, make sure to get on Sandbox and check that the restrictions are crystal clear. There was a problem not too long ago where someone posted a batch of 40 HITs for $0.01 each and the part that said to do only one was not clearly displayed. Workers who got back 39 rejections for doing more than one of these were understandably furious. (more on rejections and what they mean in (9))

For my part, I don't see the point in posting batches at all, unless you have a couple of experiments and you want to allow subjects to do them all. In that case, it can be a good way to get lots of subjects quickly for multiple experiments and even reduce the cost a bit. But if you have multiple conditions and want subjects to do just one, it will usually be easier overall and less headache for you if you write a single page that contains all of your conditions and chooses one of the conditions at random for each subject.

4. If you're posting a single HIT rather than a batch, you'll probably find that you need to pay a bit more for the HIT. Finding your HIT requires some investment of time on workers' part, and workers won't bother doing a HIT with a very low rate (e.g. $.01) if there's only one of them. If there's a batch, they're more likely, but make sure to heed the warning in (3c) above!
5. You want to make workers as comfortable as possible so it's good to use a progress bar to indicate how far into the experiment they are. An example of an experiment that makes use of a nice simple progress bar is http://kevinleung.com/psych/adjectives/ . You should be able to steal the code for it from that page.
6. You might also consider including a brief survey at the end for age, gender, how they were doing the task, and comments. For language tasks ask about first language.
7. The HIT settings can be important: Restrict to workers with a good approval rate (I use 95%), and to workers in the US. Submiterator can do this for you. Put a time limit on the experiment (so that people don't leave and come back), but don't make it too short.
8. Posting HITs on weekdays in the early morning, say 7AM, seems to yield the best results. If you post in the evening or weekend, you may get funny results. According to Chris Potts, who's done tons of mTurk stuff, this is because the best (least rushed) workers are bored office workers who are Turking to get double pay, and stay-at-home moms. I've also heard it suggested that this is because some Indian workers use fake US IP addresses, so even if you restrict to US only you'll find that all of your work has been done between 1AM and 5AM. Another possibility is that the kind of people who are on mTurk in the middle of the night are weird. Either way, don't do it.
9. When you've got some results, you have to choose to accept or reject each assignment using the web interface. This will be either at https://requester.mturk.com/batches (if you posted a batch) or at https://requester.mturk.com/mturk/manageHITs (if you posted a single HIT). Your workers expect for you to accept or reject them quickly, at most within 1 day of them completing the assignment and preferably sooner. If you're posting a big experiment, DON'T wait for all of the data to come in before

doing this, or your workers will get annoyed and you'll get a bad rep.
10. Rejecting an assignment is not something you should do lightly. It reduces workers' acceptance rate, which can affect them negatively since it's very common to restrict HITs to workers with approval rates ≥ 95%. This can mean that they have trouble getting work, which is bad for them since many of them rely on Turking as a main source of income. They will then, in turn, get on Turkernation and scream about being mistreated by you, and then no one will do your experiments. So, definitely feel free to reject or block workers if they're obviously screwing around, but if it looks like they didn't understand the task or something like that, it's potentially better for your reputation and future experiments (not to mention nicer) to just pay them anyway and then exclude their data.
10. You can either download your results in JSON format from the command line (details in the Submiterator README file) or as a .csv from https://requester.mturk.com/mturk/manageHITs . How your data comes back will depend on how precisely you passed it to mmturkey and from there to mTurk in your JavaScript. I'm in the habit of writing a little python script for each experiment that parses out my results into a .csv that looks how I want. However, if you're really careful about how you pass your data to mTurk you may be able to skip this step, depending on what you want to collect. Fortunately you don't need to wait until your first experiment is done to see how this works: just post your experiment to the Sandbox and do it yourself (or better, have some friends do it and give you feedback). Then download the results and see if it's something you can work with. If not, look back at your JavaScript and see if you can improve the way you're keeping track of data, and iterate.

Payment

You can calculate the "effective hourly rate" of your experiment by figuring out how long it will take to do it. My impression from reading Turkernation posts is that most U.S.-based Turkers will not do a HIT with an effective hourly rate of less than \$3 (cost of living and all), and anything above \$6 is considered a godsend.

That said, the choice of how much to pay is more complicated than just calculating effective hourly rate. To take an example, suppose you have a since HIT (not in a batch) that takes 30 seconds to complete and pays \$.03. The effective hourly rate is \$6, so this seems like a pretty good deal. But it's not, because Turkers have to go to the effort of finding your single HIT and when they're done they have to find another HIT to do, just three pennies richer. For them, if would be much better to have a longer HIT with the same effective hourly rate or even a slightly lower one, since then they're not wasting time searching for work. This dynamic is not as bad if you're posting a batch, since they can then simply run through the batch and do your tiny HITs one after another. (However, see (3c) in the hints section above for warnings about posting batches!)

There's research that suggests that how much you pay will not dramatically affect the quality of your results (http://homepage.psy.utexas.edu/homepage/students/buhrmester /BuhrmesterKwangGosling_PoPS_inpress.PDF). However, paying too little will affect how many subjects you get, and how long it takes to get them. If you post an experiment and no one is taking it, you can take it down and re-post with a slight increase in pay, say $.10, and see if that helps.

Noah suggests that costs may have been increasing recently. This is consistent with my impression as well. If I had to guess, I would blame two things: the increasing sophistication of the informal self-organization of Turkers via Turkernation and Turkopticon, and the rising popularity of mTurk as a tool for doing experiments. On the latter point, it's increasingly common to encounter people paying $1 for a 5-minute political science or social psychology survey. In any case, be prepared to recalibrate your payments from time to time as Turkers' expectations may change, and you may find from time to time that an experiment that you think pays just fine doesn't get subjects until you take it down and re-post with a higher rate.

Bonus/bonusing workers

For an automated way of bonusing workers, check this (https://github.com/desmond-ong/amtBonusScript) out. Instructions are in the file; basically all you need is a .csv file with the assignmentID, workerID and bonus amount.

Further information

Check out http://experimentalturk.wordpress.com/ for a lot of links to papers discussion pros and cons of using mTurk for data collection, and other useful stuff.

Getting your results

- As mentioned above, you can download your results from the Amazon web interface, but the problem with that is that it would usually mess up the order of your result variables, and it will have labels like "Answer 1", "Answer 2", instead of your variable names (Bad Amazon.).

- Using submiterator, you can download your results in JSON format, and your variables will be ordered nicely. You can use a simple python script to parse the JSON into .csv [todo: put up a sample python script and sample data file].

- Depending on your needs, your script could parse the JSON into a "wide-format" file with each worker as one row, and then read that into R (or your favourite stats software). Some people find it easier to use "long-format" in R, where each response to each question is one row -- you can do that within the python script, or you can also do that within R (using say, melt() in library(reshape2)).

- I highly recommend trying to do this while your experiment is in the sandbox. While in the sandbox, try doing your experiment, and then try downloading your data and processing it. This could give you feedback on better ways to manage your data in your code, help you iron out your parsing script and experimental flow, and importantly, make sure that you are storing all the variables that you might ever need (for example, certain random choices that you made during your experiment, presentation order, etc).

Discussion space for experiments and payment

Please post here links to your experiments, short descriptions, how much you paid, and any hurdles you encountered or other notable features. I'll start:

Property Induction experiment: ~10 minutes. I think I paid $.30 initially and was not getting subjects, so I bumped it up to $.40 or something like that and didn't have trouble after that. Got help from a positive review on Turkernation.

http://www.stanford.edu/~danlass/experiment/animals.html

Generics experiment: ~5 minutes, but the graphics part only worked in Firefox and Opera so I had to restrict it to these browsers. Because of browser restrictions I was losing most of my potential subjects when I offered $.25. To get subjects I had to pay a lot more, $.50 I think. Once I did this someone posted a link to the experiment on Turkernation (as a "piggy bank survey"), and I got lots of subjects in no time. http://www.stanford.edu/~danlass/experiment/rocks.html

Gradable adjectives experiment: Subjects hated this one. I tried out various rates and ended up with $.60, which is a lot.

http://www.stanford.edu/~danlass/experiment/w-tags.html

- Some additional notes on installation, in case you have Java-related trouble installing CLT:

I initially had trouble getting CLT to work due to weird stuff with the version of Java I have installed; eventually I managed to get it working by finding a legacy version of Java buried somewhere in my library and setting my JAVA_HOME environment variable to this. Don't know if others will have this problem, but I wanted to flag this for y'all just in case. Not every version of Java will work, apparently.

Check out this link if you want to find out where your java home is:
http://developer.apple.com/library/mac/#qa/qa1170/_index.html
then you can edit the file ~/.bash (or other terminal preferences file, this works for OS X) and add the line
export JAVA_HOME=/System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home

The script at /bin/invoke.sh in the folder where you are keeping CLT assumes that your version of Java is in a folder called /bin/ that is one level deeper than your JAVA_HOME path. If this isn't right for you, you will need to modify the last line of invoke.sh in tandem with setting the JAVA_HOME path in your .profile (or whatever) to reflect the correct location.