# TurKit

## Tools for Iterative Tasks on Mechanical Turk

CSAIL

Greg Little
Lydia B. Chilton
Max Goldman
Robert C. Miller

### What is Mechanical Turk?

Amazon's Mechanical Turk (MTurk) is an increasingly popular web service for paying people to do simple human computation tasks. Workers on the system (turkers) are typically paid a few cents for Human Intelligence Tasks (HITs) that can be done in under a minute. MTurk has already been leveraged by industry and academia for image labeling, verifying addresses, and tagging documents. MTurk is quite efficient at many tasks, and can get work done that people won't do for free.

### What are Iterative Tasks?

Currently, MTurk is largely used for *independent* tasks. Task requesters post a group of HITs that can be done in parallel, such as labeling 1000 images. We consider a different model for employing turkers: **iterative tasks**, in which a succession of turkers do tasks that build each other. For example, turkers can take turns improving a passage of text; verify each other's work by voting on it; and implement the comparison function of an iterative sorting algorithm.

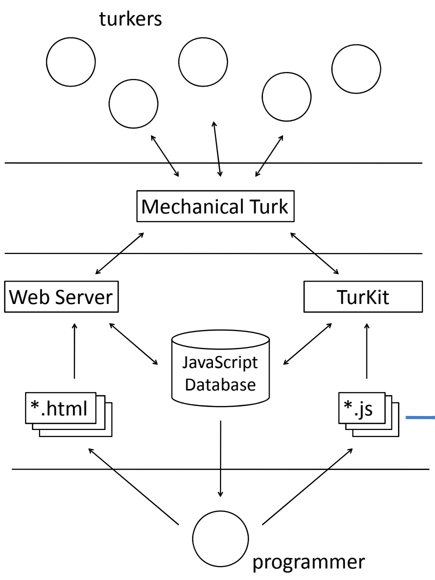
### What is TurKit?

TurKit is a toolkit for programming iterative processes on MTurk. The main challenge is making the entire system fault tolerant, so that bugs and system crashes do not lead to wasted money or time. TurKit overcomes this challenge while maintaining a straightforward procedural programming model. In this model, important function calls are performed only once, and their results are stored (memoized) in a database. As a result, a TurKit programs can be run repeatedly without redoing previously completed work.

### Future Work

For future work, we plan to explore more complicated algorithms using TurKit, such as a parallel sort algorithm that is more robust to human comparison functions that may be noisy or only partially ordered. Also valuable to users of TurKit would be a detailed study of MTurk's properties as a programming system – latency, error rate, turker expertise, etc.

### Architectual Overview



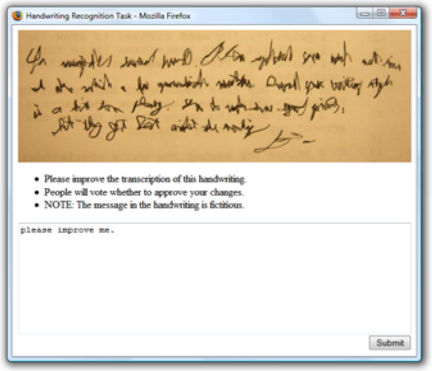
### Handwriting Recognition Example

#### TurKit Program

```
text = "please improve me."
money = $0.30

while (money > 0) {
    hit = createImproveHIT(…, text, …)
    hit = waitForHIT(hit)

    newText = hit.newText

    vote = createVoteHIT(…, text,
                                newText, …)
    vote = waitForHIT(vote)

    if (vote.result == "yes") {
        approveHIT(hit)
        money -= $0.02
        text = newText
    } else {
        rejectHIT(hit)
    }

    approveHIT(vote)
    money -= $0.01
}
```

#### Execution Log

| | Function Call | Return Value |
|---|---|---|
| 1 | createImproveHIT | hit ID "XY76532QYZZ4" |
| 2 | waitForHIT | {… newText = "You (?) (?) (?) (work). (?) (?) (?) work (not) (time). I (?) (?) a few grammatical mistakes. Overall your writing style is a bit too (phoney). You do (?) have good (points), but they got lost amidst the (writing). (signature)" …} |
| 3 | createVoteHIT | hit ID "227XAJ8299HKK" |
| 4 | waitForHIT | {… result = "yes" …} |
| 5 | approveHIT | No return value |
| 6 | approveHIT | No return value |
| 7 | createImproveHIT | hit ID "8U664982HJJ4" |
| 8 | waitForHIT | {… newText = "You (?) (?) (saved) (work). (?) (?) (?) work (not) (time). I (?) (?) a few grammatical mistakes. Overall your writing style is a bit too (phoney). You do (?) have good (points), but they got lost amidst the (writing). (signature)" …} |
| 9 | createVoteHIT | hit ID "876YHNN2JLLS" |
| 10 | waitForHIT | {… result = "no" …} |
| 11 | rejectHIT | No return value |
| 12 | approveHIT | No return value |
| 13 | createImproveHIT | hit ID "76YHHJQW129P" |
| 14 | waitForHIT | HIT not completed yet… |



- Please improve the transcription of this handwriting.
- People will vote whether to approve your changes.
- NOTE: The message in the handwriting is fictitious.

```
please improve me.
```

**Iterations**

**version 1:**
You (?) (?) (?) (work). (?) (?) (?) work (not) (time). I (?) (?) a few grammatical mistakes. Overall your writing style is a bit too (phoney). You do (?) have good (points), but they got lost amidst the (writing). (signature)

**version 2:**
~~You (?) (?) (saved) (work). (?) (?) (?) work (not) (time). I (?) (?) a few grammatical mistakes. Overall your writing style is a bit too (phoney). You do (?) have good (points), but they got lost amidst the (writing). (signature)~~

**version 3:**
You (?) (?) (**several**) (work**s**). (?) (?) (?) work (not) (time). I (?) (?) a few grammatical mistakes. Over-all your writing style is a bit too (phoney). You do **make** **some** good (points), but they got lost amidst the (writing). (signature)

**version 4:**
You (**misspelled**) (several) (**words**). (?) (?) (?) work **next** (time). I **also** **notice** a few grammatical mistakes. …

**version 5:**
You (misspelled) (several) (words). (**Plan?**) (**spell-check**) (**your**) work next time. I also notice a few grammatical mistakes. …

**version 6:**
You (misspelled) (several) (words). **Please** spellcheck your work next time. I also notice a few grammatical mistakes. Overall your writing style is a bit too phoney*. You do make some good (points), but they got* lost amidst the (writing*). (signature*)

*Red words should be *flowery*, *get*, *verbiage*, and *B-*

### Acknowledgments

This work was supported in part by the National Science Foundation under award number IIS-0447800. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsors.