Crowdsourcing and Human Computer Interaction Design

Crowdsourcing and Human Computation
Lecture 10

Instructor: Chris Callison-Burch
TA: Ellie Pavlick

Website: crowdsourcing-class.org
Wizard of Oz in HCl
Wizard of Oz in HCl
AI is lacking compared to human intelligence. Some people earn a living as "ractors", interacting with customers in virtual reality entertainments. Ractors are more expensive than AI, so the only reason to use them is because customers can tell the difference. Virtual reality entertainment has become one ongoing Turing Test, and software is continuously failing it.
Wizard of Turk?

- Can we make SciFi a reality with crowdsourcing?
- Last week we examined the possibility of using humans as a function call in TurKit.
- Can we use people in next generation interfaces for computers and mobile devices?
- What challenges does that present?
Word Processing: Boring HCI?

• Word processing supports a complex cognitive activity

• Writing is difficult: even experts routinely make style, grammar and spelling mistakes.

• Decisions like changing from past to present tense, or cutting 1/2 a page require many transformations across a document

• Current software provides little support for such tasks
Soylent: A Word Processor with a Crowd Inside

- Use large crowd of editors ala Wikipedia to improve your own work
- Use people’s basic knowledge of English to edit the document to fix errors
- Opens up many other possibilities:
  - scan for superfluous words to trim
  - update addresses with zip codes
  - do things that Word cannot (false positives in spell check)
Soylent: A Word Processor with a Crowd Inside

- Implemented as a plugin to Microsoft Word using Microsoft Visual Studio Tools for Office (VSTO)
- Makes calls to Amazon Mechanical Turk with TurKit
- Has a set of 3 special purpose modules designed for work processing
  - Shortn
  - CrowdProof
  - The Human Macro
Shortn

- A text shortening service that cuts selected text down to 85% of its original length typically without changing the meaning of the text or introducing errors.
(Aside: Motivation for compression)

- Tweets are 140 characters
- Short URLs are ~20 characters
- Image descriptions target ~120 characters
REFERENCES


AI approaches

• Rewriting text to be shorter is a task that Natural Language Processing researcher work on – including me and my students!

• The goal of “sentence compression” is to re-write text to be shorter while preserving all of its meaning
AI approaches

- Deletion
- Paraphrasing
- Summarization
AI approaches

Congressional leaders reached a last-gasp agreement Friday to avert a shutdown of the federal government, after days of haggling and tense hours of brinksmanship.
Congressional leaders reached a last-gasp agreement Friday to avert a shutdown of the federal government, after days of haggling and tense hours of brinksmanship.
Congress agreed Friday to avert a shutdown of the federal government, after days of haggling and tense hours of brinksmanship.
Congressional leaders reached a last-gasp agreement Friday to avert a shutdown of the federal government, after days of haggling and tense hours of brinksmanship.
Shortn Interaction

- Selects the paragraph or section of text that is too long
- Press the Shortn button in the Word’s Soylent ribbon tab
- Soylent launches a series of MTurk Turk tasks and notifies user when text is ready
- User launches the Shortn dialog box
Automatic clustering generally helps separate different kinds of records that need to be edited differently, but it isn't perfect. Sometimes it creates more clusters than needed, because the differences in structure aren't important to the user's particular editing task. For example, if the user only needs to edit near the end of each line, then differences at the start of the line are largely irrelevant, and it isn't necessary to split base on those differences. Conversely, sometimes the clustering isn't fine enough, leaving heterogeneous clusters that must be edited one line at a time. One solution to this problem would be to let the user rearrange the clustering manually, perhaps using drag-and-drop to merge and split clusters. Clustering and selection generalization would also be improved by recognizing common test structure like URLs, filenames, email addresses, dates, times, etc.
Automatic clustering generally helps separate different kinds of records that need to be edited differently, but it isn't perfect. Sometimes it creates more clusters than needed, because the differences in structure aren't important to the user's particular editing task. For example, if the user only needs to edit near the end of each line, then differences at the start of the line are largely irrelevant, and it isn't necessary to split based on those differences. Conversely, sometimes the clustering isn't fine enough, leaving heterogeneous clusters that must be edited one line at a time. One solution to this problem would be to let the user rearrange the clustering manually, perhaps using drag-and-drop to merge and split clusters. Clustering and selection generalization would also be improved by recognizing common test structure like URLs, filenames, email addresses, dates, times, etc.
Automatic clustering generally helps separate different kinds of records that need to be edited differently, but it isn't perfect. Sometimes it creates more clusters than needed, because the differences in structure aren't important to the user's particular editing task. For example, if the user only needs to edit near the end of each line, then differences at the start of the line are largely irrelevant, and it isn't necessary to split base on those differences.

Conversely, sometimes the clustering isn't fine enough, leaving heterogeneous clusters that must be edited one line at a time. One solution to this problem would be to let the user rearrange the clustering manually, perhaps using drag-and-drop to merge and split clusters. Clustering and selection generalization would also be improved by recognizing common test structure like URLs, filenames, email addresses, dates, times, etc.
Automatic clustering generally helps separate different kinds of records that need to be edited differently, but it isn't perfect. Sometimes it creates more clusters than needed, because the differences in structure aren't important to the user's particular editing task. For example, if the user only needs to edit near the end of each line, then differences at the start of the line are largely irrelevant, and it isn't necessary to split base on those differences. Conversely, sometimes the clustering isn't fine enough, leaving heterogeneous clusters that must be edited one line at a time. One solution to this problem would be to let the user rearrange the clustering manually, perhaps using drag-and-drop to merge and split clusters. Clustering and selection generalization would also be improved by recognizing common test structure like URLs, filenames, email addresses, dates, times, etc.
Length reduction

- Reductions affect different parts of the text, so moving slider changes different regions.
- Removes ~15–30% in a single pass, and up to ~50% with multiple iterations.
- The algorithm preserves meaning, cutting only unnecessary language and repetitions.
- User (not Workers) must remove whole arguments or sections.
Print publishers are in a tizzy over Apple’s new iPad because they hope to finally be able to charge for their digital editions. But in order to get people to pay for their magazine and newspaper apps, they are going to have to offer something different that readers cannot get at the newsstand or on the open Web.
The metaDESK effort is part of the larger Tangible Bits project. The Tangible Bits vision paper, which introduced the metaDESK along with and two companion platforms, the transBOARD and ambientROOM.

| 7 paragraphs | Reduced to 87% length of original | $7.45 264 workers | 49–84 min per paragraph |
In this paper we argue that it is possible and desirable to combine the easy input affordances of text with the powerful retrieval and visualization capabilities of graphical applications. We present WenSo, a tool that uses lightweight text input to capture richly structured information for later retrieval and navigation in a graphical environment.

<table>
<thead>
<tr>
<th>5 paragraphs</th>
<th>Reduced to 90% length of original</th>
<th>$7.47</th>
<th>52–72 min per paragraph</th>
</tr>
</thead>
<tbody>
<tr>
<td>23 sentences</td>
<td>652 words</td>
<td>284 workers</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3 shows the pseudocode that implements this design for Lookup. FAWN-DS extracts two fields from the 160-bit key: the low order bits of the key (the index bits) and the next 15 low order bits (the key fragment).
CrowdProof

While GUIs made computers more intuitive and easier to learn, they didn't let people be able to control computers efficiently.

While GUIs made computers more intuitive and easier to learn, they didn't let people be able to control computers efficiently.

• A human-powered spelling and grammar checker that finds problems Word misses, explains the problems, and suggests fixes.
Challenges for Soylent?

• In Soylent, Turkers are directly editing your documents

• What are the major concerns when other people are editing your documents?
High variance in user contributions

- Lazy workers – some workers do as little work as necessary to get paid
- Eager beavers – some do too much work or give random things that we didn’t ask for
Lazy worker

The theme of loneliness features throughout many scenes in Of Mice and Men and is often the dominant theme of sections during this story. This theme occurs during many circumstances but is not present from start to finish. In my mind for a theme to be pervasive is must be present during every element of the story. There are many themes that are present most of the way through such as sacrifice, friendship and comradeship. But in my opinion there is only one theme that is present from beginning to end, this theme is pursuit of dreams.
Eager Beaver

The theme of loneliness features throughout many scenes in Of Mice and Men and is often the principal, significant, primary, preeminent, prevailing, foremost, essential, crucial, vital, critical theme of sections during this story.
QC is hard

Insurance company may use the information to raise rates or to deny the insurance. Insurance company may use the information to raise rates or to deny the insurance. Insurance company may use the information to raise rates or to deny the insurance. Insurance companies may use the information to raise rates or to deny the insurance.

<table>
<thead>
<tr>
<th>Original</th>
<th>For <strong>serendipity</strong> discovery, the time taken is considered short.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold</td>
<td>For <strong>serendipitous</strong> discovery, the time taken is considered short.</td>
</tr>
<tr>
<td>distance = 33</td>
<td><strong>Serendipitous discoveries do not take long.</strong></td>
</tr>
<tr>
<td>distance = 3</td>
<td><strong>For serendipity discovery, the time taken is considered short.</strong></td>
</tr>
</tbody>
</table>
The find-fix-verify pattern

- No clear way to embed gold standard control data into tasks of this type
- Find-fix-verify is a 3 step process to try to ensure higher quality results
- Meant to correct the imbalance of work between lazy workers and eager beavers, and to reduce introduction of errors
Step 1: Find

- Identify passages that need improvement
- For proofreading: find at least 1 phrase or sentence that needs to be edited
- Aggregate across many independent opinions
- Regions with agreement are more likely to be correctable
Step 2: Fix

- Send the selected regions to other Worker to correct
- Each task now consists of a constrained edit to an area of interest
- Workers can see the whole paragraph but only edit the selected region
- 3-5 workers suggest alternate edits
Step 3: Verify

• Verify is a mechanism for performing quality control on the suggested edits

• Randomize the order of the proposed changes, and ask other Turkers to vote on the best one, or to flag poor suggestions

• Exclude workers who proposed the fixes, so they can’t vote on their own work
Why use find-fix-verify?

• Why should tasks be split into independent Find-Fix-Verify stages?

• Why not let Turkers fix errors they find?

• Wouldn’t that be more efficient and cost effective?

• Does it solve problems with lazy workers? How?
Cost of find-fix-verify

<table>
<thead>
<tr>
<th></th>
<th>Shortn</th>
<th>Crowdproof</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find</td>
<td>$0.55</td>
<td>$0.06</td>
</tr>
<tr>
<td>Fix</td>
<td>$0.48</td>
<td>$0.08</td>
</tr>
<tr>
<td>Verify</td>
<td>$0.38</td>
<td>$0.04</td>
</tr>
<tr>
<td>Total</td>
<td>$1.41</td>
<td>$0.18</td>
</tr>
</tbody>
</table>

per paragraph  per error
However, while GUI made using computers be more intuitive and easier to learn, it didn’t let people be able to control computers efficiently. The masses only can use the software developed by software companies, unless they know how to write programs.
This is an argument about whether there should be a standard “NoSQL storage” API to protect developers storing their stuff in proprietary services in the cloud. Probably unrealistic. To protect yourself, use an open software offering, and self-host or go with hosting solution that uses open offering.
The Human Macro

• Macros usually require users to translate their intentions into algorithms explicitly via a scripting language

• The human macro is a “Natural Language Crowd Scripting Language”

• It allows the user to ask other people complete tasks like formatting citations or finding appropriate figures
Like Siri but unrestricted

- Natural language interfaces still struggle with unconstrained input
- Humans are good at understanding written instructions
The Human Macro

The Human Macro

Title
Find Creative Commons figure for paragraph

Create Task for Every:
Paragraph

Instructions (with Example)
I need a creative commons licensed image to describe under Creative Commons.

Mechanical Turk Worker Preview

Advertisement

Find Creative Commons figure for paragraph

Instructions
I need a creative commons licensed image to describe under Creative Commons.

Here is the text:
When I first visited Yosemite State Park in California, the rocks were big, the trees were big, the animals were big, and the granite mountain that looks like it was sheared off.
Design challenges

• Ensure that the user creates tasks that are scoped correctly for a Mechanical Turk worker

• Ask user provide an example input and output, to clarify task requirements

• Prevent the user from spending money on a buggy command

• The Human Macro helps debug the task by allowing a test run on a sentence or paragraph
Showing the results

• User specifies if Turkers’ work should replace the existing text or just annotate it
• If replace, text is underlined with drop-down substitution
• If annotate, feedback is inserted in comment bubbles anchored to selected text using Word’s comments interface
<table>
<thead>
<tr>
<th>Request</th>
<th>“Please change text in document from past tense to present tense.”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>I gave one final glance around before descending from the barrow. As I did so, my eye caught something [...]</td>
</tr>
<tr>
<td>Output</td>
<td>I give one final glance around before descending from the barrow. As I do so, my eye catches something [...]</td>
</tr>
<tr>
<td>Request</td>
<td>“Pick out keywords from the paragraph like Yosemite, rock, half dome, park. Go to a site which has CC licensed images [...]”</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Input</td>
<td>When I first visited Yosemite State Park in California, I was a boy. I was amazed by how big everything was [...]</td>
</tr>
<tr>
<td>Output</td>
<td><img src="image-url" alt="Image of Yosemite Park" /></td>
</tr>
</tbody>
</table>
### Human Macro Examples

<table>
<thead>
<tr>
<th>Request</th>
<th>“Please find the bibtex references for the 3 papers in brackets. You can located these by Google Scholar searches and clicking on bibtex.”</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
<td>Duncan and Watts [Duncan and watts HCOMP 09 anchoring] found that Turkers will do more work when you pay more, but that the quality is no higher.</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>@conference{ title={Financial incentives and [...]}}, author={Mason, W. and Watts, D.J.}, booktitle={HCOMP ‘09}</td>
</tr>
</tbody>
</table>
Human Macro Examples

<table>
<thead>
<tr>
<th>Request</th>
<th>“Please complete the addresses below to include all information needed as in example below. [...]”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Max Marcus, 3416 colfax ave east, 80206</td>
</tr>
</tbody>
</table>
| Output  | Max Marcus  
3416 E Colfax Ave  
Denver, CO 80206 |
Soylent’s contributions

• The idea of embedding paid crowd workers in an interactive user interface to support complex cognition and manipulation tasks on demand

• Crowd workers can do HCI tasks that computers cannot reliably do automatically

• Easier to ask workers to do something than it is to write macro script
This paper presents Soylent, a word processing interface that uses crowd workers to help with proofreading, document shortening, editing and commenting tasks. Soylent is an example of a new kind of interactive user interface in which the end user has direct access to a crowd of workers for assistance with tasks that require human attention and common sense. Implementing these kinds of interfaces requires new software programming patterns for interface software, since crowds behave differently than computer systems. We have introduced one important pattern, FindFix-Verify, which splits complex editing tasks into a series of identification, generation, and verification stages that use independent agreement and voting to produce reliable results. We evaluated Soylent with a range of editing tasks, finding and correcting 82% of grammar errors when combined with automatic checking, shortening text to approximately 85% of original length per iteration, and executing a variety of human macros successfully.
Would you let just anyone edit your documents?

• Quality – do you believe that they are doing what we ask?

• Accuracy – do we have safeguards in place to avoid workers introducing errors?

• Privacy – do we trust them with the material? Is it sensitive?
Would you let them read your email?

EmailValet displays assistants that create a task stream from emails in the user's inbox. They are given limited, accountable access to the user's inbox so that they can bury overwhelming volume of incoming requests. New messages push important requests down by new messages, become hard to find or triaging is error prone. To avoid overlooking important messages, people often keep a cluttered inbox power of their email. However, they may extract tasks from each email. Task management directly integrated into the email client draws on crowdsourced expert assistants to transform a large, human effort into the email client. Automatic techniques have shown some promise in identifying tasks in emails yet fully reliable and/or expensive.

Current approaches for handling email are often mixed with other emails, get pushed down by new messages, become hard to find, and require heavy manual handling user interface. EmailValet introduces techniques that define control for users. As a result of their email management: assistants are willing to share with assistants that define a subset of their inbox that they are willing to share with assistants. They have access to crowdsourced assistants that carry out tasks when users specify rules. To support accountability and/or a fee, participants are given limited, accountable access to the user's inbox so that they can recruit remote assistants from powered systems that define alert for tasks when a user's inbox is cluttered. EmailValet is an implicit part of every inbox, but the users are comfortable sharing their inbox with assistants. EmailValet's assistants would you let them read your email?