My research combines computation with the intelligence of crowds—large groups of people connecting and coordinating online—to create hybrid human-computer systems. Combining machine and crowd intelligence opens up a broad new class of software systems that can solve problems neither could solve alone. However, while crowds are increasingly adept at straightforward parallel tasks, they struggle to accomplish many goals: participants vary in quality, well-intentioned contributions can introduce errors, and future participants amplify and propagate those errors. Moreover, crowds that can generate the necessary information might not even exist yet, or their knowledge might be distributed across the web.

My work in human-computer interaction develops computational techniques to overcome these limits in crowdsourcing. These solutions adopt a broad view of how we might integrate crowds and computation: 1) embedding crowd work into interactive systems, 2) creating new crowds by designing social computing systems, and 3) mining crowd data for interactive applications.

CROWD-POWERED SYSTEMS

I create systems that are powered by crowd intelligence: these systems advance crowdsourcing from a batch platform to one that is interactive and realtime. To embed crowds as first-order building blocks in software, we need to address problems in crowd work quality and latency. This research develops computational techniques that decompose complex tasks into simpler, verifiable steps, return results in seconds, and open up a new design space of interactive systems.

To instantiate these ideas, I developed a crowd-powered word processor called *Soylent* that uses paid micro-contributions to aid writing tasks such as text shortening and proofreading [1]. Using Soylent is like having an entire editorial staff available as you write. For example, crowds shorten the user's writing by finding wordy text and offering alternatives that the user might not have considered. The user selects from these alternatives using a slider that specifies the desired text length (Figure 1). On average, Soylent can shorten text up to 85% of its original length. By using tighter wording rather than wholesale cuts, the system shortens a ten-page draft by a page and a half in only a few minutes. Soylent also offers human-powered proofreading and natural-language macros.

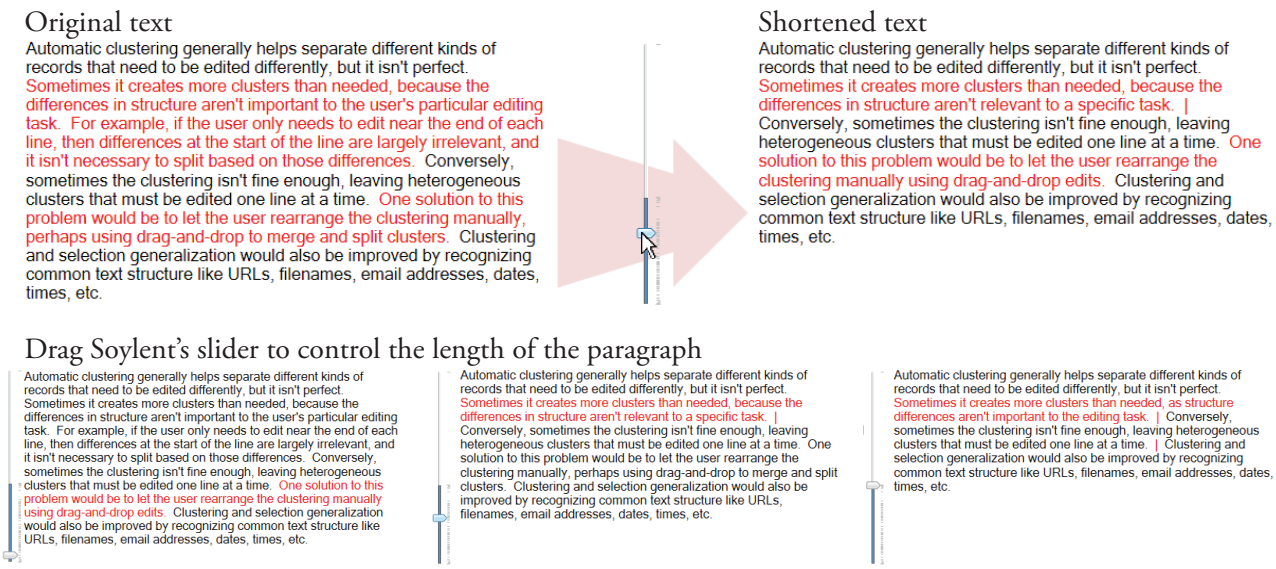

Figure 1. Soylent combines contributions from paid crowd workers to generate a large number of alternative shortenings for a document. The user drags the slider to specify the desired text length. Red text indicates locations where cuts or rewrites have occurred.

To improve Soylent's crowd work quality, I developed a design pattern called *Find-Fix-Verify* that guides workers through open-ended editing processes (Figure 2). Traditional approaches that ask workers on Mechanical Turk to shorten or proofread text do not work: lazy workers all focus on the simplest problem and overeager workers introduce undesired changes. Find-Fix-Verify splits complex tasks into creation and review stages that use independent agreement to produce reliable results.

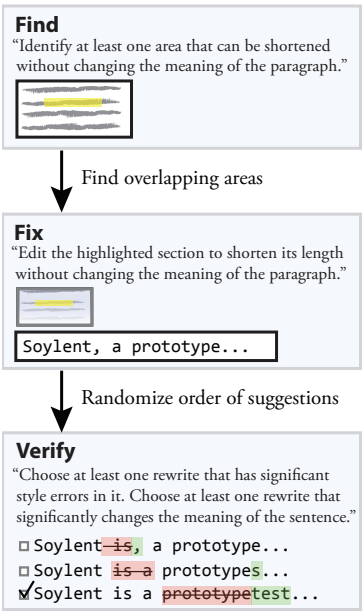

Figure 2. Find-Fix-Verify improves work quality for open-ended tasks like text shortening.

This work was awarded Best Student Paper at ACM UIST 2010, and has been cited over fifty times in the year since it was published. Find-Fix-Verify has already been adapted by researchers for tasks like image segmentation, map labeling, and formal crowd programming languages. The work has also been included in the curricula of several graduate human-computer interaction and crowdsourcing classes, including at Stanford University, UC Berkeley, UT Austin, University of Michigan, University of Toronto, University of Maryland at College Park, UC Santa Cruz, and Harvard University.

Many crowd-powered systems need responses in seconds, not minutes. For example, digital camera users want to preview their photos immediately, before typical crowds could finish any work. In response, I developed *Adrenaline*, which can recruit workers in two seconds and execute large searches in ten seconds [2]. This realtime crowdsourcing can complete typical tasks such as votes in just five seconds, and can also power complex interactive systems such as digital cameras. An Adrenaline camera captures a short video instead of one frame, then uses the crowd to decide on the best moment (Figure 3). This camera can identify the best smile, catch subjects in mid-air jumps, and decide on the best angle available, all in seconds.

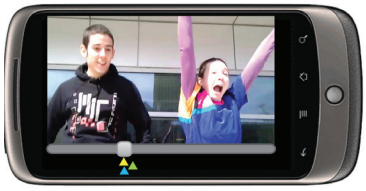
Figure 3. Adrenaline's realtime crowds operate a camera shutter by choosing the most photographic moment seconds after it happens.

Adrenaline's recruitment approach, called the *retainer model*, pays workers a small wage to be on call and respond quickly when asked. Its insight is that algorithms can guide realtime crowds to complete tasks more quickly than even the fastest individual crowd member. The Adrenaline camera thus focuses workers' attention on areas where they are likely to agree, reducing a large search space to a final answer in a few seconds.

CREATING CROWDS: THE DESIGN OF SOCIAL SYSTEMS

When existing crowds are not appropriate for the goal, I design social systems that create new crowds. I pair these designs with studies of online communities.

I create crowds that collect information unknown to most people. For example, I introduced the concept of *friendsourcing,* which collects information known only to members of a social network [3]. *Collabio*, my friendsourced person-tagging application, gathered over 29,000 friend-authored tags on thousands of individuals on Facebook [4]. Evaluations demonstrated that the vast majority of this information is not available elsewhere on the web. Collabio tags have already been used to create personalized news feeds, question-routing systems, and social network exploration tools. Friendsourcing applications can also amplify existing social interactions. I developed a friendsourced blog reader called *FeedMe* that helps active web readers share interesting content, then observes those shares to learn recipient interest models and improve its sharing recommendations [5]. In effect, FeedMe is a social collaborative filtering engine. Other researchers have also applied friendsourcing for applications such as adding edges to social network graphs.

I complement these systems with empirical work studying existing crowds and communities. For example, researchers and practitioners often assume that strong user identity and permanent archives are critical to online communities. However, communities like 4chan /b/ attract millions of users without identity systems or archives. Our investigation of five million posts on /b/ was the first large-scale analysis of an anonymous, ephemeral social system [6]. We found that /b/ moves incredibly quickly: the median thread spends just five seconds on the first page and less than five minutes on the site before new content replaces it. Over 90% of posts are made by fully anonymous users, with other identity signals adopted and discarded at will. The lack of user identity and archives produces a fast-churning ecosystem that encourages experimentation and likely plays a strong role in /b/'s ability to create well-known memes and drive internet culture.

Our work on 4chan /b/ was awarded Best Paper at AAAI ICWSM 2011, and an online video of my conference talk has been viewed over one thousand times.

INTERACTIVE CROWD DATA

When crowds browse the web or use social media, they leave activity traces that can be mined for interactive applications. I build user experiences that utilize this data, as well as tools that allow end users to directly explore it. These goals often create opportunities for algorithms in data mining and information retrieval.

Query logs offer an opportunity to rethink not just search ranking, but the entire search user experience. Inspired by this goal, I created *Tail Answers,* which return direct results for thousands of information needs like conference submission deadlines, the average body temperature for dogs (Figure 4), and molasses substitutes [7]. I mine search logs to find common session endpoints for informational queries, then use paid crowds to extract the answer from endpoint webpages. Tail Answers significantly improve subjective ratings of search quality and users' ability to solve their information needs without clicking on a result.

Figure 4. Tail Answers mine browser logs to generate thousands of direct answers to queries.

By analyzing social data feeds, applications can also help users make sense of their network and unfolding events. I developed a Twitter topic browser called *Eddi* that introduces a web search technique for mining open-world topics out of microblog updates. Using this TweeTopic algorithm, Eddi users can filter and explore their feed via implicit topics like research, C++ or a favorite celebrity [8]. TweeTopic outperforms existing topic modeling approaches for Twitter browsing and the interface doubles users' ability to find worthwhile content.

RESEARCH AGENDA

I aim to make crowd computing, web-scale data, and social networks integral parts of computation. Soylent is typical of my approach: I pick a goal that is beyond the capabilities of a crowd platform, understand why the system has those limits, push forward the discipline by developing techniques to overcome those limits, and embed the ideas in an application that achieves my original goal. This section outlines some future opportunities that I am excited to pursue.

What is the crowdsourcing platform of the future? While Amazon Mechanical Turk is a useful prototyping platform, its current incarnation has serious limitations. For example, it will be critical to move toward more expert work. I am interested in platforms that could assemble flash crowds of experts to pipeline large tasks. Starting with a sketch for a user interface, such a platform could find a designer to create a mock-up, pass the mock-up to a usability professional for testing, loop until the design meets usability goals a few hours later, and finally recruit a programmer to implement the interface. Put another way: what would it take to crowdsource a presentation, or an entire software program, or a symphony?

A science of crowdsourcing. There has been explosive growth in crowdsourcing over the past two years, but we need to pair discovery with principles. Specifically, we need to develop design patterns and best practices for crowd computing. We also lack methods to analyze and compare the complexity of crowd computing algorithms. A formal framework or set of benchmarks would allow us to compare approaches along axes such as cost and size of crowd, latency, quality and worker stress.

Social application design. I have a long-standing interest in using social and crowd computing to create new user interface tools. I want users to be able to draw on thousands of professional authors' styles to improve their own writing; to galvanize gigabytes of open-source code to auto-complete not just the line of Python they are writing now, but the entire design pattern they are trying to apply; to mine social network status update feeds to personalize search; to accelerate navigation through a popular but poorly-designed web site.

In graduate school, I have been fortunate to collaborate with over fifty coauthors and ten academic departments. I intend to continue this tradition of collaboration as I move forward.

In sum, my research draws on crowds to innovate in human-computer interaction, and it draws on human-computer interaction to innovate in crowd computing. This work opens opportunities for a broad class of systems that combine human and machine intelligence.

REFERENCES

[1] **Michael S. Bernstein**, Greg Little, Robert C. Miller, Björn Hartmann, Mark Ackerman, David R. Karger, David Crowell, and Katrina Panovich. Soylent: A word processor with a crowd inside. In Proceedings of UIST 2010: *ACM Symposium on User Interface Software and Technology*. New York, New York, 2010.
Best Student Paper Award.

[2] **Michael S. Bernstein**, Joel Brandt, Robert C. Miller, and David R. Karger. Crowds in two seconds: Enabling real-time crowd-powered interfaces. In Proceedings of ACM UIST 2011: *ACM Symposium on User Interface Software and Technology*. Santa Barbara, California, 2011.

[3] **Michael S. Bernstein**, Desney Tan, Greg Smith, Mary Czerwinski, and Eric Horvitz. Personalization via friend-sourcing. ACM Transactions on Computer-Human Interaction 17(2), 2010.

[4] **Michael S. Bernstein**, Desney Tan, Greg Smith, Mary Czerwinski, and Eric Horvitz. Collabio: A game for annotating people within social networks. In Proceedings of UIST 2009: *ACM Symposium on User Interface Software and Technology*. Victoria, British Columbia, 2009.

[5] **Michael S. Bernstein**, Adam Marcus, David R. Karger, and Robert C. Miller. Enhancing directed content sharing on the web. In Proceedings of CHI 2010: *ACM Conference on Human Factors in Computing Systems*. Atlanta, Georgia, 2010.

[6] **Michael S. Bernstein**, Andrés Monroy-Hernández, Drew Harry, Paul André, Katrina Panovich, and Greg Vargas. 4chan and /b/: An analysis of anonymity and ephemerality in a large online community. In Proceedings of ICWSM 2011: *AAAI Conference on Weblogs and Social Media*. Barcelona, Spain, 2011.
Best Paper Award.

[7] **Michael S. Bernstein**, Jaime Teevan, Dan Liebling, Susan Dumais, and Eric Horvitz. Direct results for search queries in the long tail. In Proceedings of CHI 2012: *ACM Conference on Human Factors in Computing Systems*. Austin, Texas, 2012.

[8] **Michael S. Bernstein**, Bongwon Suh, Lichan Hong, Jilin Chen, Sanjay Kairam, and Ed H. Chi. Eddi: Interactive Topic-Based Browsing of Social Status Streams. In Proceedings of UIST 2010: *ACM Symposium on User Interface Software and Technology*. New York, New York, 2010.