Crowdsourcing for (almost) Real-time Question Answering

Denis Savenkov
Emory University
dsavenk@emory.edu

Scott Weitzner
Emory University
sweitzn@emory.edu

Eugene Agichtein
Emory University
eugene@mathcs.emory.edu

Abstract

Modern search engines have made dramatic progress in the answering of many user's questions about facts, such as those that might be retrieved or directly inferred from a knowledge base. However, many other questions that real users ask are more complex, such as asking for opinions or advice for a particular situation, and are still largely beyond the competence of the computer systems. As conversational agents become more popular, QA systems are increasingly expected to handle such complex questions, and to do so in (nearly) real-time, as the searcher is unlikely to wait longer than a minute or two for an answer. One way to overcome some of the challenges in complex question answering is crowdsourcing. We explore two ways crowdsourcing can assist a question answering system that operates in (near) real time: by providing answer validation, which could be used to filter or re-rank the candidate answers, and by creating the answer candidates directly. Specifically, we focus on understanding the effects of time restrictions in the near real-time QA setting. Our experiments show that even within a one minute time limit, crowd workers can produce reliable ratings for up to three answer candidates, and generate answers that are better than an average automated system from the LiveQA 2015 shared task. Our findings can be useful for developing hybrid human-computer systems for automatic question answering and conversational agents.

1 Introduction

It has long been a dream to communicate with a computer as one might with another human being using natural language speech and text. Nowadays, we are coming closer to this dream, as natural language interfaces become increasingly popular. Our phones are already reasonably good at recognizing speech, and personal assistants, such as Apple Siri, Google Now, Microsoft Cortana, Amazon Alexa, etc., help us with everyday tasks and answer some of our questions. Chat bots are arguably considered "the next big thing", and a number of startups developing this kind of technology has emerged in Silicon Valley and around the world[1].

Question answering is one of the major components of such personal assistants. Existing techniques already allow users to get direct answers to their factoid questions. However, there is still a large number of more complex questions, such as advice or accepted general opinions, for which users have to dig into the "10 blue links" and extract or synthesize answers from information buried within the retrieved documents. To cater to these informational needs, community question answering (CQA) sites emerged, such as Yahoo! Answers and Stack Exchange. These sites provide a popular way to connect information seekers with answerers. Unfortunately, it can take minutes or hours, and sometimes days, for the community to respond, and some questions are left unanswered altogether.

To facilitate research on this challenge, TREC

[1] http://time.com/4194063/chatbots-facebook-messenger-kik-wechat/

LiveQA shared task[2] was started in 2015, where automated systems attempt to answer real users' questions within a 1 minute period. This task was successful, with the winning system able to automatically return a reasonable answer to more than half of the submitted questions, as assessed for TREC by the trained judges from NIST. Nevertheless, many questions were unable to be answered well by any of the participating systems.

In this work we explore two ways common crowdsourcing can be used to help an automated system answer complex user questions in near real-time scenario, e.g., within a minute. More specifically, we study if crowd workers can quickly and reliably judge the quality of the proposed answer candidates, and if it is possible to obtain reasonable written answers from the crowd within a limited amount of time. Our research questions can be stated as:

1. RQ1. Can crowdsourcing be used to judge the quality of answers to non-factoid questions under a time limit?
2. RQ2. Is it possible to use crowdsourcing to collect answers to real user questions under a time limit?
3. RQ3. How does the quality of crowdsourced answers to non-factoid questions compare to original CQA answers, and to automatic answers from TREC LiveQA systems?

2 Methodology

To answer the research questions, we conducted a series of crowdsourcing experiments using the Amazon Mechanical Turk platform[3]. We used questions from the TREC LiveQA 2015 shared task, along with the system answers, rated by the NIST assessors[4]. The questions for the task were selected by the organizers from the live stream of questions posted to the Yahoo! Answers CQA platform on the day of the challenge (August 31, 2015). For these questions we also crawled their community answers, that were eventually posted on Yahoo! Answers[5].

To check if crowdsourcing can be used to judge the quality of answers under a time limit (RQ1), we asked workers to rate answers to a sample of 100 questions using the official TREC rating scale:

1. Bad - contains no useful information
2. Fair - marginally useful information
3. Good - partially answers the question
4. Excellent - fully answers the question

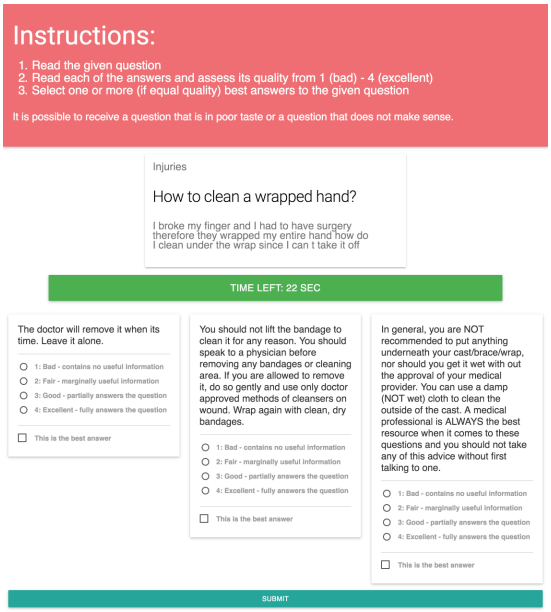

Figure 1: Answer validation form

We chose to display 3 answers for a question, which were generated by three of the top-10 automatic systems from TREC LiveQA 2015 evaluation (Agichtein et al., 2015). To study the effect of time pressure on the quality of judgments we split participants into two groups. One group made their assessments with a 1 minute countdown timer shown to them, while the other could complete the task without worrying about a time limit. Within each group, we assigned three different workers per question, and the workers were compensated at a rate of $0.05 per question for this task.

The interface for collecting answer ratings is illustrated in Figure 1[6]. On top of the interface workers were shown the instructions on the task, and question and answers were hidden at this time. They

[2] www.trec-liveqa.org

[3] http://mturk.com

[4] https://sites.google.com/site/trecliveqa2016/liveqa-qrels-2015

[5] As the answer we took the top question, which was selected as the "Best answer" by the author of the question or by the community.

[6] The screenshots show the final state of the form, as we describe later in this sections fields were unhidden step-by-step for proper timing of reading, answering and validation

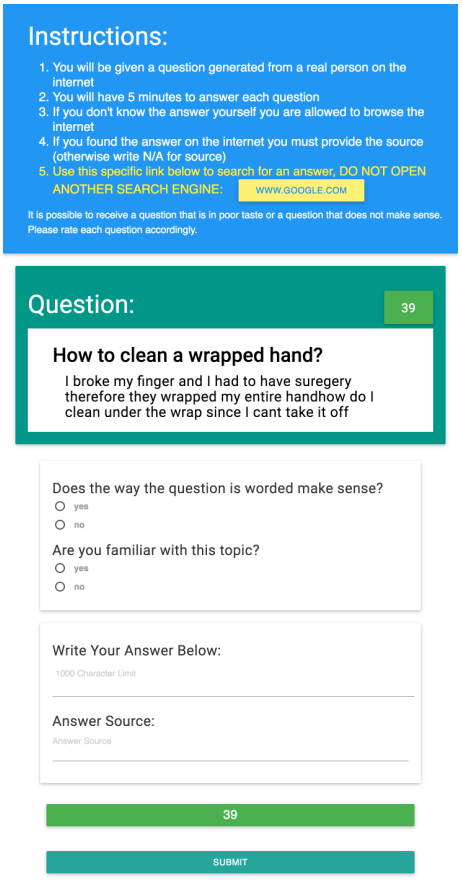

Figure 2: Answer crowdsourcing form

were instructed to read the question, read the answers, and rate each answer's quality on a scale from 1 (Bad) to 4 (Excellent), and finally choose a subset of candidates that best answer the question. Upon clicking a button to indicate that they were done reading the instructions, the question, a 60 second countdown timer and 3 answers to the question appeared on the screen. At the 15 second mark the timer color changed from green to red. In the experiments without time pressure the timer was hidden, but we still tracked the time it took for the workers to complete the task.

In another experiment, designed to answer RQ2 and check whether crowd workers can provide an answer to a given question within a limited amount of time, we asked different workers to answer the questions from TREC LiveQA 2015. We split the workers into two groups and displayed a one minute countdown timer for one of them. We left a grace period and let the workers submit their answers after the timer had run out. The workers received a $0.10 compensation for each answer. The form for answer crowdsourcing is shown in Figure 2, and similar to the answer rating form, it starts with a set of instructions for the task. We let the users browse the internet if they were not familiar with the topic or could not answer the question themselves. To prevent them from finding the original question on Yahoo! Answers, we included a link to Google search engine with a date filter enabled[7]. Using this link, workers could search the web as it was on 8/30/2015, before TREC LiveQA 2015 questions were posted and therefore workers were in the same conditions as automatic systems on the day of challenge[8]. Initially, the question was hidden for proper accounting of question-reading and answering times. Upon clicking a button to indicate that they were done reading the instructions, a question appeared along with a button, which needed to be clicked to indicate that they were done reading the question. After that, the answering form appears, it contained four fields:

1. Does the question make sense: "yes" or "no" to see if the question was comprehensible
2. Are you familiar with the topic: A yes or no question to evaluate whether the worker has had prior knowledge regarding the question topic
3. Answer: the field to be used for the user's answer to the given question
4. Source: the source used to find the answer: URL of a webpage or NA if the worker used his own expertise

Finally, to compare the quality of the collected answers with automatic system and CQA responses (RQ3) we pooled together the crowdsourced answers, the answers from the winning and other top-10 LiveQA'15 systems, and the original answers crawled from Yahoo! Answers. Each set of answers was given to mechanical turk workers for ratings.

3 Results and Discussion

In this section we will describe our results and discuss some of the implications. We start from the results on answer rating (Section 3.1), and then describe the answer crowdsourcing experiment (Section 3.2).

[7]https://www.google.com/webhp?tbs=cdr:1,cd_max:8/30/2015

[8]The ranking of search results could be different on the day of the challenge and for our workers

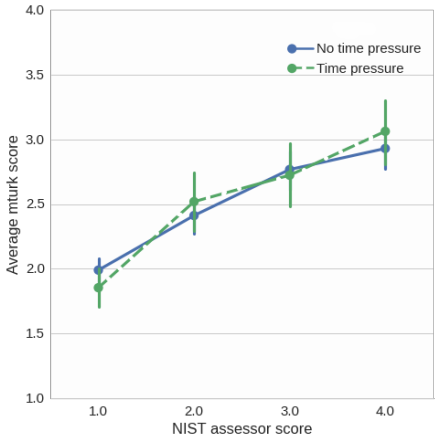

Figure 3: Correlation between NIST assessor scores and crowdsourced ratings with and without time limit on the work time

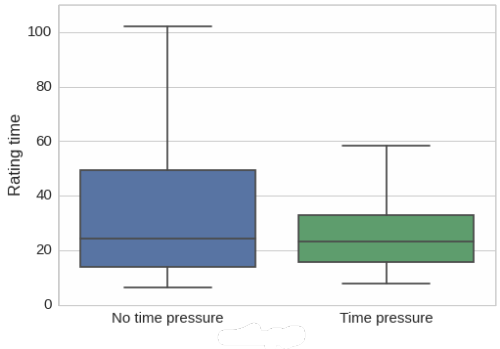

Figure 4: Box plot of answer rating time with and without time pressure

3.1 Answer rating

In the answer rating experiment we collected 6 ratings (3 with and 3 without time pressure) for each of three answers for a sample of 100 questions, which makes it a total of 1800 judgments. Each answer also has an official NIST assessor rating on the same scale. Figure 3 shows correlation between official NIST assessor relevance judgments and ratings provided by our workers. The Pearson correlation between the scores is $\rho = 0.52$. The distribution of scores shows that official assessors were very strict and assigned many extreme scores of 1 or 4, whereas mechanical turk workers preferred intermediate 2s and 3s. The results did not show any significant differences between experiments with and without time pressure. Figure 4 shows that even though the median time to rate all three answers is around 22-25 seconds in both experiments, the upper bound is significantly lower in the experiment with the time pressure.

Therefore, we conclude that in general we can trust crowdsourced ratings, and on average one minute is enough to judge the quality of three answers to CQA questions.

3.2 Answer crowdsourcing

In the answer crowdsourcing experiment we collected 6 answers (3 with and without time pressure) for each of the 1087 LiveQA'15 questions. Since we have answers from different sources, let's introduce the following notations:

- *Yahoo! Answers* - answers eventually posted by users on Yahoo! Answers for the original questions
- *Crowd* - answers collected from Mechanical Turk workers without time pressure
- *Crowd-time* - answers collected from Mechanical Turk workers with one minute time pressure
- *LiveQA winner* - answers from the TREC LiveQA'15 winning system
- *LiveQA top10* - answers from another top 10 TREC LiveQA'15 system.

Table 1 summarizes some statistics on the answers. The first thing to notice is that, unlike CQA websites, where some questions are left unanswered, by paying the crowd workers we were able to get at least one answer for all LiveQA questions (after filtering "No answer" and "I don't know" kind of responses). The length of the answers, provided by Mechanical turk users is lower, and time pressure forces users to be even more concise. The majority of workers ($\sim 90\%$) didn't use the web search and provided answers based on their experience, opinions and common knowledge.

From Figure 5 we can see that adding time pressure shifts the distribution of answering times[9]. The tail of longer work times for no time limit experiment becomes thin with time restrictions and the distribution peaks around one minute.

To estimate the quality of answers, we took a sample of 100 questions and repeated the answer rating

[9]We had separate timers for reading the instructions, the question, and writing the answer, the inclusion of instruction-reading time is why the total time could be more than 1 minute

Table 1: Statistics of different types of answers for Yahoo! Answers questions

Statistic	Y!A	mTurk	mTurk-time	LiveQA'15 winning system
% answered	78.7%	100.0%	100.0%	97.8%
Length (chars)	354.96	190.83	126.65	790.41
Length (words)	64.54	34.16	22.82	137.23

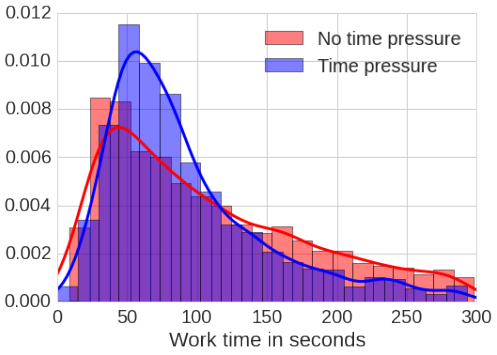

Figure 5: Distribution of answering times for experiments with and without time pressure

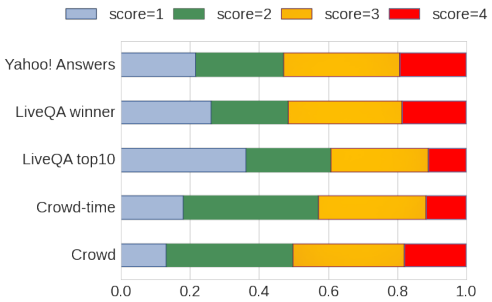

Figure 7: Distribution of scores for different types of answers to Yahoo! Answers questions

experiment on this data. Each answer was judged by 3 different workers (without time pressure), and their scores were averaged. Figure 6 displays the plot with average score for answers from different sources. Quite surprisingly the quality of collected answers turned out be comparable to those of CQA website users. Average rating of answers produced by the winning TREC LiveQA system is also pretty close to human answers. Finally, as expected, time pressure had its negative effect on the quality, however it is still significantly better than quality of an average top 10 QA system.

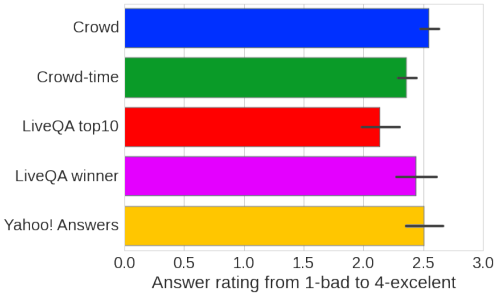

Figure 6: Average scores of different types of answers to Yahoo! Answers questions

Analysis of the score distribution (Figure 7) sheds some light on the nature of the problems with automatic and human answers. The automatic systems generate non-relevant answers ($score = 1$) more often than human, either because the systems fail to retrieve relevant information, or to distinguish between useful and non-useful answer candidates. However, by having a larger information store, e.g., the Web, automated QA systems can often find a perfect answer ($score = 4$), while crowd workers tend to give generally useful, but less perfect responses ($score = 2, 3$).

Our results suggest that the "crowd" can quickly give a reasonable answer to most CQA questions. However, some questions require a certain expertise, which a common crowd worker might not possess. One idea to tackle this challenge is to design a QA information support system, which a worker can use to help them find additional information. For example, in our experiment, we let workers use web search to find answers, if they were unfamiliar with the topic; more effective search interfaces may be helpful.

Overall, the results of our study are encouraging, but there are a number of questions, that need to be addressed in order to build an efficient real-time human-computer QA system, which we leave to future work. For example, in order for a hybrid crowd-automatic QA system to scale to a large number user questions, an efficient crowd gathering and job assignment components are required(Bernstein et al., 2011). One potential approach to scalability is selective crowdsourcing, i.e., that a system consult

the crowd only if it does not believe it has a good candidate, or cannot decide between multiple good answers based on internal quality scores. Another promising direction is for a QA system to use the crowd data as a feedback loop, and improve its performance over time, e.g., to update the answer selection and ranking models.

4 Related Work

Using the wisdom of a crowd to help users satisfy their information needs has been studied before in the literature. Bernstein et al. (2012) explored the use of crowdsourcing for offline preparation of answers to tail search queries. In this work log mining techniques were used to identify potential question-answer fragment pairs, which were then processed by the crowd to generate the final answer. This offline procedure allows a search engine to increase the coverage of direct answers to user questions. In our work, however, the focus is on online question answering, which requires fast responses to the user, who is unlikely to wait more than a minute. Another related work is targeting a different domain, namely SQL queries. The CrowdDB system (Franklin et al., 2011) is an SQL-like processing system for queries, that cannot be answered by machines only. In CrowdDB human input is used to collect missing data, perform computationally difficult functions or matching against the query. In Aydin et al. (2014) authors explored efficient ways to combine human input for multiple choice questions from the "Who wants to be a millionaire?" TV show. In this scenario going with the majority for complex questions isn't effective, and certain answerer confidence weighting schemas can improve the results.

Using crowdsourcing for relevance judgments has been studied extensively in the information retrieval community, e.g., (Alonso et al., 2008; Alonso and Baeza-Yates, 2011; Grady and Lease, 2010) to name a few. The focus in these works is on document relevance, and the quality of crowdsourced judgments. Whereas in our paper we are investigating the ability of a crowd to quickly assess the quality of the answers in a nearly real-time setting.

Crowdsourcing is usually associated with offline data collection, which requires significant amount of time. Its application to (near) real-time scenarios poses certain additional challenges. Bernstein et al. (2011) introduced the retainer model for recruiting synchronous crowds for interactive real-time tasks and showed their effectiveness on the best single image and creative generation tasks. We are planning to build on these ideas and integrate a crowd into a real-time question answering system. The work of Lasecki et al. (2013) showed how multiple workers can sit behind a conversational agent named Chorus, where human input is used to propose and vote on responses. Another use of a crowd for maintaining a dialog is presented in Bessho et al. (2012), who let the crowd handle difficult cases, when a system was not able to automatically retrieve a good response from the database of twitter data. In this paper, we focus on a single part of the human-computer dialog, i.e. question answering, which requires a system to provide some useful information in a response to the user.

5 Conclusions

We explored the potential usefulness of crowdsourcing for near real-time question answering, by either directly collecting answers from the crowd, or by using crowdsourced judgments to quickly validate automated answers. Our initial results show that crowd workers are capable of validating a small set of answer candidates quickly, which could be potentially incorporated into an automatic QA system for answer validation and reranking (RQ1). In addition, even one minute appears enough for a crowd to generate a fair or good response to most real questions drawn from a CQA site, which can be useful in case a QA system didn't have good candidates in the first place (RQ2). Finally, we compared crowdsourced answers to the original Yahoo! Answers responses, and to the responses of purely automated LiveQA'15 systems (RQ3). The quality of crowdsourced answers was comparable to the original Yahoo! Answers responses, and even with time pressure, crowdsourcing was shown promising to complement or augment automated QA systems.

ACKNOWLEDGMENTS: This work was partially supported by the Yahoo Labs Faculty Research Engagement Program (FREP).

References

Eugene Agichtein, David Carmel, Donna Harman, Dan Pelleg, and Yuval Pinter. 2015. Overview of the trec 2015 liveqa track. In *Proceedings of TREC 2015*.

Omar Alonso and Ricardo Baeza-Yates. 2011. Design and implementation of relevance assessments using crowdsourcing. In *Advances in information retrieval*, pages 153–164. Springer.

Omar Alonso, Daniel E. Rose, and Benjamin Stewart. 2008. Crowdsourcing for relevance evaluation. *SIGIR Forum*, 42(2):9–15, November.

Bahadir Ismail Aydin, Yavuz Selim Yilmaz, Yaliang Li, Qi Li, Jing Gao, and Murat Demirbas. 2014. Crowdsourcing for multiple-choice question answering. In *AAAI*, pages 2946–2953.

Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. 2011. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 33–42. ACM.

Michael S Bernstein, Jaime Teevan, Susan Dumais, Daniel Liebling, and Eric Horvitz. 2012. Direct answers for search queries in the long tail. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 237–246. ACM.

Fumihiro Bessho, Tatsuya Harada, and Yasuo Kuniyoshi. 2012. Dialog system using real-time crowdsourcing and twitter large-scale corpus. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '12, pages 227–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael J Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. 2011. Crowddb: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 61–72. ACM.

Catherine Grady and Matthew Lease. 2010. Crowdsourcing document relevance assessment with mechanical turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's mechanical turk*, pages 172–179. Association for Computational Linguistics.

Walter S. Lasecki, Rachel Wesley, Jeffrey Nichols, Anand Kulkarni, James F. Allen, and Jeffrey P. Bigham. 2013. Chorus: A crowd-powered conversational assistant. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 151–162, New York, NY, USA. ACM.