SocialKeyboard: Proofreading Everyday Writings in Mobile Phones

Jin-woo Lee
Department of Knowledge Service Engineering,
KAIST, Daejeon, Korea
jinwoo.lee@kaist.ac.kr

Jae-Gil Lee
Department of Knowledge Service Engineering,
KAIST, Daejeon, Korea
jaegil@kaist.ac.kr

Joohyun Kim
Department of Knowledge Service Engineering,
KAIST, Daejeon, Korea
joohyun.kim@kaist.ac.kr

Uichin Lee
Department of Knowledge Service Engineering,
KAIST, Daejeon, Korea
uclee@kaist.ac.kr

CHI'15 Extended Abstracts, Apr 18-23, 2015, Seoul, Republic of Korea
ACM 978-1-4503-3146-3/15/04.
http://dx.doi.org/10.1145/2702613.2732901

Abstract
The flood tide of professional proofreading services has exceeded the market demand, but many pioneering competitors seem to be torn between quality and cost. Furthermore, proofreading of everyday writings has received little attention thus far, which is important for non-native speakers. In this paper, we propose SocialKeyboard that uses mobile crowd workers that can be contacted in a mobile environment. By embedding proofreading features into existing mobile keyboards, SocialKeyboard also achieves easy access anytime and everywhere in mobile environments. Moreover, it aims to induce faster responses by implementing 1) Push based task assignment to mobile crowd workers, 2) Real-time track changes, and 3) Synchronous task chaining of proofreading and verification. We present a first working prototype of SocialKeyboard and describe the interplay between Requester, Proofreader, and Verifier. Our preliminary evaluation showed promising results in terms of task speed, and we plan to perform a large-scale study in the near future.

Author Keywords
SocialKeyboard; crowdsourcing; human computation; proofreading

ACM Classification Keywords
H.5.2 [User Interfaces]: Input devices and strategies; I.3.6 [Methodology and Techniques]: Interaction techniques

Introduction

Abundance of rising startup companies in the online proofreading industry indicates a strong demand in this field of services. Among these companies, ChattingCat recently won the NUVC B2C competition [1], gaining recognition for its proofreading service value. Traditionally, proofreading was offered through a professional offline setting. Most proofreading for academic papers is still offered through language centers of schools or private services. Because of hiring costs and limited availability of proofreaders, usages of these offline resources are usually time and money consuming for quality results. Therefore, trivial and everyday writings, such as emails and text messages among friends, are rarely proofread, despite its importance for non-native speakers' language learning. In reality, only writings of importance, such as academic papers and books, are professionally proofread.

Thus, we wanted to investigate the online proofreading services for everyday lifestyle writing that provide high accessibility and fast responses at a relatively low cost. Our work focuses on the mobile setting in order to reach everyday lives of non-native speakers across all ages. Since our primary goal for this service is to deliver an inexpensive and prompt service to mobile users, we considered an approach of crowdsourcing, rather than outsourcing.

We propose SocialKeyboard, a crowdsourcing based proofreading service for mobile devices. SocialKeyboard augments an existing mobile keyboard and allows users to proofread any texts entered via mobile keyboards. Users can immediately send a proofread request for any texts using an existing mobile keyboard. To achieve fast responses, SocialKeyboard utilizes push-based task assignment to mobile crowd workers. Requesters can track changes in real-time for better language learning. SocialKeyboard can maintain quality by employing multiple proofreads and including verification steps.

Related Work

Each company uses slightly different approaches and solutions to provide online proofreading services for non-native speakers. These approaches can be grouped into two major solutions: i.e., one with an automated computer dependent service using natural language processing, Ginger for example [5], and the other solution involving connection with real tutors who are available 24/7. Both time and price of service depends on how much of human workforce is used.

Soylent is a crowd-powered proofreading service for Microsoft Word by leveraging Amazon Mechanical Turk (M-Turk) work force [3]. It allows users to send tasks, such as shortening, proofreading, and macro editing, in large scale of writing. It uses a Find-Fix-Verify procedure that splits each task into multiple stages to preserve work quality and control costs. However, this pattern of work faces a major latency problem as the time lag cutoff line is 15 minutes, meaning that it actually takes longer to wait for another Turker to agree on each Find, Fix, or Verify stage, compared to the time it takes for Turker to actually perform the

given task. These delays make Soylent incompatible as a real time service model.

ChattingCat uses a designated web interface through which users can send proofreading requests and receive responses [4]. Unlike Soylent, a request is sent to a single worker, and it only supports request resending for quality assurance. Our work considers proofreading of everyday writings in mobile environments. We tap into existing mobile keyboards and text input interfaces to allow users to directly send proofreading requests and to track changes in real-time.

Ginger uses a 100% computer generated algorithm to automatically provide a proofreading service [5]. However, in general, its accuracy and quality of service is inferior to human-assisted proofreading.

quikTurkit considers a method of providing real-time service requirements in M-Turk by maintaining a pool of workers online [6]. Unlike quikTurkit, which is based on pull-based M-Turk services, we use push-based mobile crowdsourcing methods to satisfy real-time requirements.

SocialKeyboard

SocialKeyboard is a service that supports a proofreading service in a mobile environment by pushing proofread requests to mobile crowd workers. The proofreading task consists of two micro-tasks, i.e., proofreading of original sentences and verification of revised sentences. Note that there are three user roles in our system: i.e., Requester, Proofreader, and Verifier.

Our work uses a simple Fix-Verify (or Proofread-Verify) procedure as opposed to Find-Fix-Verify in Soylent [3]. We use a two-stage process, and as shown later, support real-time track changes, thereby significantly lowering overall proofreading latency.

The key distinction of SocialKeyboard from existing proofreading services is that it directly uses mobile keyboards to allow users easy access to proofreading services at any usage context of mobile applications. Furthermore, it uses 1) push based task assignment, 2) real-time track changes, and 3) synchronous task chaining, to elicit faster proofreading. In the section below, we discuss each technique, and detail design rationales.

One of the key barriers in mobile environments is its inconvenience of text input and editing. While ChattingCat can be used in mobile environments, users need to switch back and forth between multiple apps (e.g., copying and pasting sentences). For example, if the user wishes proofreading on an email draft over Gmail, the text must be copied and pasted into a separate web interface, revised, and be moved back into the original email application; this is a very laborious task in smartphones. In contrast, SocialKeyboard directly uses existing mobile keyboard layouts by simply customizing an input interface that Android EditText view exists, and this improved user interface obviates the need of such text moving steps. What is important here is that Requesters can receive proofreading services without any interruption of the main task at hand (no major context switching is required).

Push based task assignment to mobile crowd workers

When designing an online community, either pull model or push model can be considered [7]. Some portion of

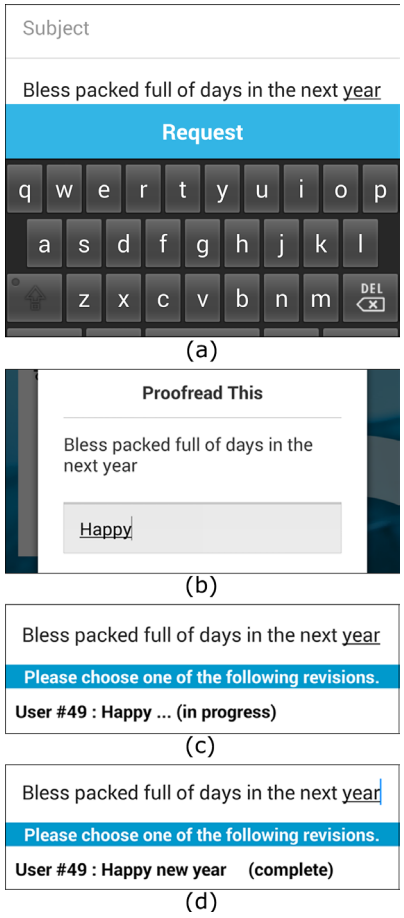

Figure 1. Screenshots of proofreading process.
(a) Requester requests in Gmail app.
(b) Proofreader 1 starts typing.
(c) Requester gets a real-time feedback.
(d) Requester gets a complete answer.

diligent workers process most of the work quota for both pull and push model based mobile labor markets. However, in the case of a pull model, it is impossible to select those hard workers among the pool of workers. In contrast, for a push model, it is possible to identify *the super agents* who achieve higher work performance. Distinguishing and making use of those super agents will lead to a high-quality proofreading service that is cheaper and faster [2]. Thus, we consider a mobile crowdsourcing environment based on the push model in which a pool of workers is maintained. Android's Google Cloud Messaging framework [8] is used to push requests directly to those super agents. In SocialKeyboard, we additionally maintain profile information of those mobile crowd workers (e.g., age, education, acceptance rate, etc.) for quality control and reputation management.

Overview of the proofreading sequence
When a Requester submits original sentences to be proofread via requests edits through SocialKeyboard, the system selects three workers at a time, from the list of Proofreaders, and this process repeats until two workers are successfully recruited. When a Proofreader declines the request or 45-second timeout occurs, the request will be redirected to three other workers from the pool. If we find two workers who have accepted the request, the requests sent to the other workers will be cancelled and a thank you message will be sent to the Proofreaders. When all Proofreaders are unavailable, the request will remain unanswered, and a sorry message will be eventually delivered to the Requester. Currently, each parameter follows the figures presented in the synchronous Q&A research [9]; however, further research is required in accordance with the conditions of a proofreading.

Since we strive to provide service at a minimum cost, it is better if fewer workers are involved in a task. At the same time, we have to prevent unconstructive or malicious Proofreaders from easily earning money while creating low quality results. For these reasons, we hire two proofreaders per task.

Real-time track changes
In a real-time Q&A system, the Requester can make better decisions if there is detailed feedback about the question that the Requester has submitted [9]. Thus, through tight coupling between Requester and Proofreader, much real-time information such as expected recruitment time of crowd workers, expected completion time of proofreading tasks, or expected result quality, could be provided prior to or during the SocialKeyboard proofreading process.

As Figure 1 shows, as soon as the Proofreaders start making corrections, the revision processes are displayed in real-time in the Requester's screen. The Requester can track how the corrections are being made. During the editing process, the Requester's screen displays that the revision is 'in progress'. This indicator then turns to 'complete' when a Responder finishes editing.

Synchronous verification
We assume that the Requester does not have the ability to pick the best-revised sentence. To solve this problem, we hire a separate Verifier to check the quality of the received results. The Proofreader's work is not directly evaluated by the Requester, but rather by the Verifier. This extra step requires closer task synchronization when chaining the process. If push notifications are sent to a Proofreader and a Verifier

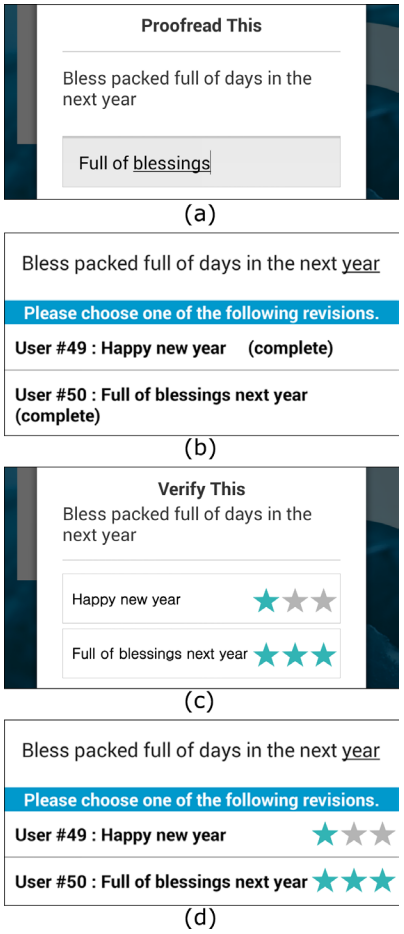

Figure 3. Screenshots of verification process.
(a) Proofreader 2 completes typing.
(b) Requester gets 2nd feedback.
(c) Verifier rates what was proofread by Responder 1 and 2.
(d) Ratings are forwarded to Requester

simultaneously, the Verifier has to wait until proofreading is finished. If the Verifier is notified after the proofreading is completed, the Requester has to wait for the verification process. Thus, the Verifier has to be notified in between requesting and proofreading process, and this notification timing was determined by subtracting the expected recruitment time of the Verifier from the completion time of proofreading tasks. Detailed point of notification is shown in Figure 2 below.

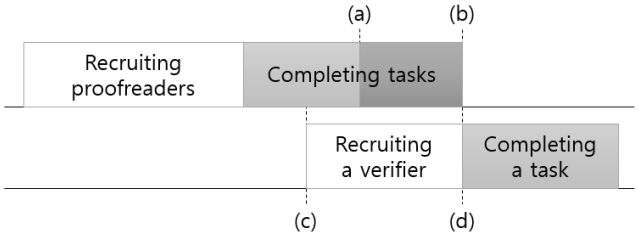

Figure 2. Sequence diagram of proofreading and verification synchronization. (a) Proofreader 1 completes. (b) Proofreader 2 completes. (c) Expected completion time of proofreading tasks – Expected recruitment time of a Verifier. (d) Expected completion time of proofreading tasks.

Similar to the Proofreader hiring process, we maintain the candidate list for Verifier recruitment. However, because the work quality was shown to be better in mobile crowdsourcing [2], we chose to hire only one Verifier. Instead, if the Requester finds a wrongful action of the Verifier, the Requester will have ability to report the problem and be compensated through the SocialKeyboard account management platform. Various strategies for preserving quality of work may exist, but when both feedbacks from two Proofreaders are poor, the Verify step is less meaningful. Thus, we use a 3-scale rating system—high, medium, and low quality—for absolute assessment. Figure 3 shows the verification process, and how these results are reflected in the Requester's screen. Finally, when the Requester clicks one of the results from the list, the original written sentences written by the Requester is replaced with the edited sentences from the Proofreaders.

Preliminary Evaluation

Experiment Design: We conducted an initial evaluation of how fast SocialKeyboard responds to a proofreading request. 21 people were hired for the reward of 10 cents for each revision, same price as the fastest service ChattingCat. 20 is the minimum number of random candidate user pool suggested in the synchronous Q&A study [9]. To remove demographic variance, the recruitments were chosen to be in their 20s and hold college degrees or advanced degrees. 20 Korean sentences were collected from Lang-8—a website in which foreigners ask for proofreading [10]. Questions were sent to the Proofreaders in a random order over 2 days. 2 out of 21 users were also randomly selected as proofreaders since the system has no clue to guess the user availability. Workers were told to proofread anytime at their convenience. Our experiments can be considered as a Korean proofreading service for non-native Korean speakers.

Results: Average completion time of two proofreading subtasks was about 170 seconds. Although push notification to crowd workers were used, overall proofreading latency was in the order of a few minutes due to the fact that not every users are available to respond immediately. Further research on parameter selection (e.g., the number of proofreaders) and user profiling techniques is needed in the future. Verification subtasks took only about 26 seconds. Even after verification, the overall waiting time was less than

that of ChattingCat, which can be interpreted as fairly encouraging results. Further research on task assignment, quality management, and usability/UX of SocialKeyboard is required.

Conclusion and Future Work

We presented a working prototype of SocialKeyboard for making everyday writings easier and quicker, through the integration of human computation power inside an on-screen mobile keyboard. By using mobile crowdsourcing, we aim to achieve delay requirement and cost effectiveness. SocialKeyboard is not tied to a specific website or app, but augments existing mobile keyboards such that proofreading can be embedded into any existing mobile applications. Furthermore, it uses push based task assignment, real-time track changes, and synchronous task chaining to provide faster responses. The key contribution of this work is the design of a novel interface that easily connects the current soft keyboard interface to crowdsourcing.

There are serval directions for future work. Push-based task assignment needs to be further investigated. Since our goal is to lower response latency, we can profile user behavior (e.g., by using a user's calendar and analyzing a user's activity data) to find a set of users who are most likely available. As indicated in Synchronous Q&A [9] study, it is possible to help a Requester make an informed decision by providing the expected response probability, the expected response quality, and estimated number of respondents. Also, because requests will be sent based on the time or availability of the responder, the Proofreader or the Verifier will also experience fewer interruptions. Therefore, further research needs to be done how additional information can be provided to connect each task more synchronously.

Acknowledgement

This work was supported by ICT R&D program of MSIP/IITP (1391104004, Development of Device Collaborative Giga-Level Smart Cloudlet Technology).

References

[1] Northern University Venture Challenge. http://nuvc.nuisepic.com/.

[2] Musthag, M., and Ganesan, D. Labor dynamics in a mobile micro-task market. In *Proc. SIGCHI* (2013), 641-650.

[3] Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., Crowell, D., and Panovich, K. Soylent: a word processor with a crowd inside. In *Proc. UIST* (2010), 313-322.

[4] ChattingCat. https://chattingcat.com/.

[5] Ginger. http://www.gingersoftware.com/.

[6] Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., Miller, R., Tatarowicz, A., White, B., White, S. and Yeh, T. VizWiz: nearly real-time answers to visual questions. In *Proc. UIST* (2010), 333-342.

[7] Resnick, P., Konstan, J., Chen, Y., and Kraut, R. E. 6 Starting New Online Communities. Building successful online communities: Evidence-based social design. (2012), 231.

[8] Google Cloud Messaging for Android. https://developer.android.com/google/gcm/index.html/.

[9] Richardson, M., and White, R. W. Supporting synchronous social Q&A throughout the question lifecycle. In *Proc. WWW* (2011), 755-764.

[10] Lang-8. http://lang-8.com/.