The Effects of Predictive Features of Mobile Keyboards on Text Entry Speed and Errors

OHOUD ALHARBI, Simon Fraser University, Canada
WOLFGANG STUERZLINGER, Simon Fraser University, Canada
FELIX PUTZE, University of Bremen, Germany

Mobile users rely on typing assistant mechanisms such as prediction and autocorrect. Previous studies on mobile keyboards showed decreased performance for heavy use of word prediction, which identifies a need for more research to better understand the effectiveness of predictive features for different users. Our work aims at such a better understanding of user interaction with autocorrections and the prediction panel while entering text, in particular when these approaches fail. We present a crowd-sourced mobile text entry study with 170 participants. Our mobile web application simulates autocorrection and word prediction to capture user behaviours around these features. We found that using word prediction saves an average of 3.43 characters per phrase but also adds an average of two seconds compared to actually typing the word, resulting in a negative effect on text entry speed. We also identified that the time to fix wrong autocorrections is on average 5.5 seconds but that autocorrection does not have a significant effect on typing speed.

CCS Concepts: • **Human-centered computing → Keyboards**;

Keywords: Predictive text, Mobile keyboard, autocorrection, error correction, error detection, backspace

ACM Reference Format:
Ohoud Alharbi, Wolfgang Stuerzlinger, and Felix Putze. 2020. The Effects of Predictive Features of Mobile Keyboards on Text Entry Speed and Errors. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 4, ISS, Article 183 (November 2020). ACM, New York, NY. 16 pages. https://doi.org/10.1145/3427311

1 INTRODUCTION

Supported by the growth of text-based social media, texting is today considered a ubiquitous form of communication. Many people prefer sending text messages over making phone calls. Further, people communicate through tweets, Facebook posts, or many other electronic text-based channels. Thus, text entry methods should support users in transcribing their thoughts in a fast and accurate manner, especially in mobile devices.

Mobile users have strong opinions about which type of keyboard they prefer and their different, idiosyncratic ways of using them. This includes, for example, the usage of automatic word prediction and the response to autocorrection events. These opinions are often informed by their interaction with mobile systems they have used in the past. To better understand the origins of these opinions, this paper mainly focuses on an analysis of the behaviors people exhibit in mobile text entry with respect to prediction and autocorrects and the associated costs in terms of time. Word prediction could save up to 45% keystrokes in mobile keyboards [15], but this promise is rarely transferred to

Authors' addresses: Ohoud Alharbi, Simon Fraser University, Canada, oalharbi@sfu.ca; Wolfgang Stuerzlinger, Simon Fraser University, Canada, w.s@sfu.ca; Felix Putze, University of Bremen, Germany, felix.putze@uni-bremen.de.

2573-0142/2020/11-ART183 $15.00
https://doi.org/10.1145/3427311

a corresponding increase in typing speed due to the need to fix wrong predictions or due to the higher cognitive load for handling word completions [25, 41].

Several sources and dimensions of behavioral variability for typing on mobile devices have been identified through lab studies, e.g., [50]. Recent research demonstrated the need for assessing real-world variability [9, 37]. Knowing what users intended to write is impossible [14]. Thus, previous studies of text entry "in the wild" [10, 18] ignored error rates, as calculating error rates for uncontrolled text input is very challenging. Furthermore, privacy concerns limit the opportunities to collect data from personal contexts [9]. To balance the need of measuring error rates with keyboard familiarity, we implement a hybrid approach, where people type predefined, controlled phrases, using their own mobile devices. To enable us to log detailed predictions and auto-corrections, their correctness, and to address potential privacy concerns, we deliberately use predefined phrases and our own predictive system.

This paper presents an online crowd-sourcing study on the cost of using the prediction panel and autocorrection with diverse participants. We analyze metrics related to the timing of individual keystrokes through detailed client-side logging in our system. After the discussion of related work, we present our refined logging and tagging mechanism, and a main study (N = 170) to observe the effect of predictive features on typing performance.

2 BACKGROUND AND RELATED WORK

Recent studies in text entry examined the effect of keyboard layouts on typing behavior, e.g., [8, 24]. Such research typically collects data in the lab to evaluate the performance (speed, error rate) of a new design. Some aspects of users' behavior are hard to observe in lab studies, as they require a fixed setting, including sometimes even a specific posture to hold the device. Lab studies also enforce a uniform protocol, e.g., to reduce variations. Yet, there is growing desire in HCI to understand users' behavior and performance beyond the lab [22, 43]. Various methods, such as ethnography and diary studies as well as experience sampling [16] and behavior sensing and recognition [11], have facilitated the explanation of behaviors in natural settings [14]. However, such approaches are not yet widely used to understand human performance and behaviors with text entry [14]. Currently, gathering typing data outside a traditional lab experiment relies on crowd-sourcing [27] or custom mobile apps [18, 42].

2.1 Predictive Features

Error correction has been highlighted as a key challenge for text entry, as it contributes substantially to slow down real-life text entry speed [6, 19]. Errors are costly in time and effort. They also affect user perception of text entry system quality. Additionally, the visibility of suggestions can increase both perception and interaction costs, which could reduce text entry speed, e.g., [21, 40], and in some cases seems to even decrease writing accuracy [7].

The effectiveness of word prediction as a feature is unclear for mainstream mobile text entry [37], considering also that switching the attention from the keyboard and the typed text to the prediction panel can cause delays [51]. Many factors play a role in the effectiveness of the use of predictive features [26], including the efficiency of text entry method and the experience of the user [37]. On the one side, offering multiple choices for corrections can provide notable benefits, because it gives users the flexibility to deal with a wide range of needs and situations and also helps users who benefit from good modality choices [36]. In other cases, offering multiple choices for corrections might lead to less favorable results with users who make overly quick or bad choices, who then also prefer systems that offer limited interaction possibilities [21]. Still, predictive features can be beneficial as an assistive technology, e.g, [3].

Some systems provide phrase correction or completion while writing, e.g, VelociTap[55]. However, phrase correction results in corrections far from the typing position, which the user might not notice. A recent study introduces a new gaze-assisted positioning technique that allows users to edit text far from the cursor position [47]. Other systems, such as the Smart-Restorable Backspace [4] and WiseType [1], suggest text that was previously written or deleted by the user to help them recover faster from mistakes. Finally, EEG-based systems have been proposed to help with wrong predictions [38].

2.2 Text Entry Evaluation

Allowing participants to type whatever they want in user studies can confounds experimentation, as there is a lack of control for performance measurements [32]. Assessing accuracy and errors is then quite challenging since there is no source to compare the written text with [32] and users might "game" the experiment by entering gibberish or short, easy words. The standard procedure in lab-based text entry studies is to show participants predetermined phrases one at a time and ask them to transcribe them [54]. These phrases should be retrieved randomly from a set and shown one at a time [31]. Transcribing presented phrases is an appropriate protocol to ensure that participants do not create phrases that break experimental control, invalidate error measurements, or decrease reproducibility [32]. Some text entry studies even prevent the use of the mouse cursor or cursor keys during entry, allowing backspace as the only technique for correction [57]. Restricting at least some of the above factors enables accurate error rate calculation [48].

The drawback of using a transcription task is that in the real world, users rarely transcribe text, but typically compose original text [54]. On the other hand, a composition task can be slow, variable, and might not exhibit the same level of performance as a copy task. Further, the cognitive overhead associated with creating compositions can increase variance between participants and decrease the internal validity of an experiment [54].

Alternative approaches ask users to create phrases based on a prompt (composition task) or to describe images (description task), e.g., [34, 54]. However, it is challenging to calculate writing accuracy with such approaches, since there is no source to compare the written text with, i.e., no reference for accuracy measurements.

Another way to conduct text entry experiments is through games with a text input component, e.g., [44, 52]. Text Blaster [52] is a game platform for conducting both laboratory and crowd-sourced text entry experiments. The competitive nature of games encourages users to enter text both quickly and accurately [52]. However, this approach also results in a large amount of garbage data that requires manual editing and human judgment [44].

An effective approach to measure performance is to ask users to copy presented phrases as quickly and accurately as possible, e.g., [31, 59]. A previous "in the wild" study calculated text entry speeds, in words per minute, by parsing out portions of data input streams [14].

The most common approaches to calculate errors involve using a transcription task [31], where the target is well-defined, or uses the judgment of crowd workers [54]. Researchers usually calculate uncorrected error rates based on the truly intended text [48, 57]. Uncorrected errors are errors that remain in the final transcribed string. The other type is corrected errors, which are any errors that are fixed during entry. Error correction takes time and is therefore subsumed in the words-per-minute (WPM) metric [57].

3 MOTIVATION

Recent research analyzed text entry with a large sample of participants on mobile or desktop keyboards, e.g., [12, 37]. These approaches use an online web-based system but are unable to analyze word prediction. Supported by fine-grained client-side logging, our system enables the analysis

of word prediction and auto-correction events, including when predictions or auto-corrections appeared, what they were, and how long it took users to select candidate words. In other words, with our approach we can detect when a word appeared in the prediction panel and how long it took until the user tapped on it. Or how much later a user tapped on a word to edit it. This enabled us to analyze participants' interaction with system failures more accurately. We asked participants to use their own keyboard, because we wanted to avoid the overhead of them having to learn a (potentially) new layout.

Our goal of analyzing word prediction and autocorrect in more detail was also motivated by previous work [9] that identified that most (27 of 30) people use word predictions and more than half (16 of 30) use autocorrection, which means that these mechanisms are frequently used. Previous studies on mobile keyboards also identified decreased performance for heavy use of word prediction [37, 41]. A recent study [37] identified the need for more detailed analyses to better understand the effectiveness of predictive features for different users. The potential time-cost of prediction and autocorrect motivates us to analyze user interaction with autocorrection and the prediction panel during text entry in more detail, in particular when these approaches fail. We formulated the following research question to drive our work in this paper: *Are human behaviours around autocorrection and word prediction with a mobile keyboard objectively beneficial, especially when these algorithms fail?*

4 APPARATUS

Our system for data collection is a web application. We implemented the system using HTML, CSS, JavaScript, and PHP. We used an Ubuntu instance and an Apache server on Amazon EC2 to host our web application. The application provides a custom autocorrection method and prediction panel, both of which work independent of the one provided by the operating system or personal keyboard of the user. The system presents prompts with text for the user to enter and logs all occurring events at the keystroke level. Due to data logging restrictions, we permitted only Android clients in the study. We did not require users to install an application.

4.1 Instructions

At the beginning, participants needed to agree that they had read the initial instructions and to also give their consent for data collection. These initial instructions asked participants to temporarily disable the predictive features on their Android devices. Once participants agreed to participate, they were instructed on the procedure and then started the text entry tasks. The main part of the experiment showed only a single line of instruction, a presented phrase, and a textbox to input that phrase, our custom prediction bar, as well as the normal keyboard used for text entry, see Figure 1. Users needed to tap on the "Next" button to move to the next phrase, where they then also saw an up-to-date average of their speed and error rate.

We used transcription typing to measure participants' typing speed, as this approach enables us to study motor performance while excluding cognitive aspects related to the process of text generation [37].

4.2 Custom Prediction Panel and Autocorrection

To ensure that we could correctly log every action, we forced users to disable their own predictive system, including their prediction panel and autocorrection. Once a users' keyboard pops up, our web-based system adds a custom prediction panel seamlessly above their normal keyboard, modeled to closely resemble the original Android panel. Our prediction panel thus looks like it is part of the users' keyboard, i.e., seems to be a built-in feature. Similar to most other prediction panels, our prediction panel shows three candidate predictions: the originally inputted text on the left,

the most probable prediction in the center (highlighted with an underline), and the second most probable one on the right. As with other systems, users can choose the middle candidate word by pressing the space key or can tap on any other candidate word in the panel to enter it (see Figure 1). We had to use a custom prediction algorithm, as we would otherwise need access to the internals of the word prediction, which current APIs do not provide, and because web-based services for prediction exhibited a latency too high to be viable for word-level predictions. Our prediction algorithm is based on string matching with Levenshtein distance [28, 45]. We used a dictionary with the 40,000 most frequent words from project Gutenberg[1]. We classify a word as being misspelled and trigger auto-correction when the Levenshtein distance [28] between the inputted and the candidate word is less than three edits, i.e., insertions, deletions, or substitutions. In this case we replace the misspelled word with the (most likely) correct one.

Our Autocorrect and prediction methods match the behaviour of slightly older Android versions, who offer less-refined methods compared to the most recent version. This is not a major concern, as most people around the world still use older versions of Android, e.g., [33].

We verified that our prediction algorithm matches commercial systems reasonably well in term of prediction performance, layout, and position. For this, we randomly chose phrases and compared the output of our system with that of an Android 9 keyboard. We found that the intended word appeared in the prediction panel 94% of the time, while the other two predictions matched what the Android keyboard suggested in 85% of all cases. Our code will be made available in the future for open access.[2]

4.3 Data Logging

We suppress the local system's auto-correction and prediction functionality by making the text entry field a password field, where we then display the inputted text. We monitor this field for changes and detect each change through comparisons with the string in the field in the previous iteration. To detect situations when a user has not turned off their predictive or autocorrection system, we created our own heuristics scheme that compares the state of the input field before and after each keystroke. Depending on the timing of the last few characters, the length of the recently entered text, and the Levenshtein edit distance, we detect the presence of machine-generated text input and show a warning message to the user that they need to disable their predictive features. A similar approach was used by Palin et al. [37] to log predictive features on the web. While we were able to register autocorrection and word prediction events with millisecond accuracy, we match the delay for individual keystrokes observed in previous work [37].

We record each text change or touch event, which roughly corresponds to the keystroke level, with a corresponding timestamp. For each phrase, we record the following data: device orientation (portrait/landscape), presented text, typed text, the complete input stream, keystrokes per character, words per minute, and total time per phrase. Also, we log all word predictions that appeared (on a character-by-character-level), as well as all picked word predictions, autocorrections, cursor movements, and error messages that were triggered during text entry. This comprehensive logging enables us to fully replay the input of each phrase and analyze participant behaviors in the context of the current phrase.

4.4 Phrase Set

We used 30 phrases randomly selected from the Enron MobileEmail phrase set [53]. We removed any non-alphabetic characters, including punctuation, and made sure that the selected phrases

[1] https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists
[2] https://github.com/Ohoudalharbi/custom_prediction_panel

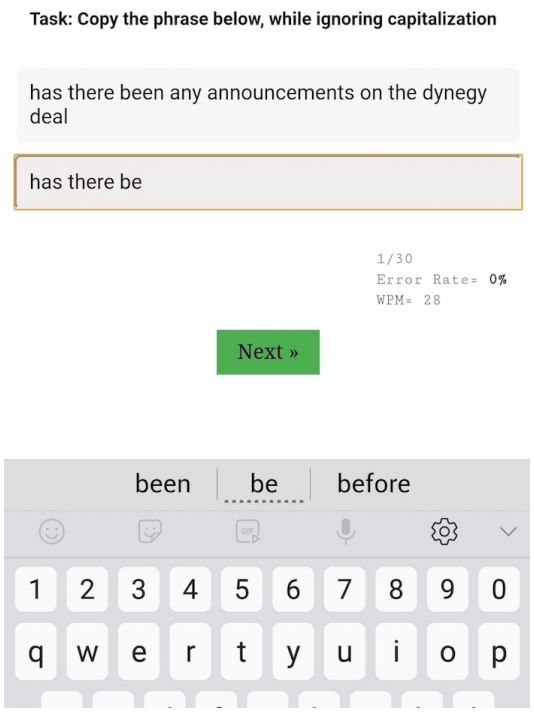


Fig. 1. The webpage that participants saw during the experiment with our custom prediction panel.

contained at least three words. The phrases in the set (774 sentences) were generally short to medium length, average 6.1 words (SD 1.68, ranging from 3 to 12), and contained on average 29.9 characters (SD 10.13, ranging from 14 to 67). The set has diverse sentences with a relatively low Out-Of-Vocabulary (OOV) rate of 1.4% with respect to the most frequent 64,000 words in the Wall Street Journal (WSJ0) corpus [53].

4.5 Task Restrictions Based on Pilots

We ran 6 pilots (lab and crowd-sourced) with a total of 112 participants. Based on observations from these pilots, we iteratively adjusted our system and instructions to account for several potentially problematic behaviors, such as the use of system-based prediction and autocorrection mechanisms mentioned above. We iteratively refined our word prediction algorithm as well as the tagging, detection, and warning mechanisms to ensure that our recorded data is as clean as possible. For example, we disabled the copy and paste feature for the field to prevent direct copying of phrases. To prevent blatant disregard of the instructions, we prevent participants from skipping phrases or sloppy writing by triggering a warning for higher than 50% average error rate.

The results of a lab pilot confirmed that the system logs all the events correctly, i.e., exactly as they appeared to participants. No noticeable delays in the appearance of word predictions or autocorrection during text entry were apparent or noticed. All editing episodes were recorded and the system also logged the time for correcting mistakes through backspace and cursor movements. The custom prediction panel integrated visually into the participants' keyboard regardless of their own keyboard outline. Our mechanism for disabling predictive features correctly worked well and

Table 1. Demographic Data

Factor	Result
Gender	Male → 56%; Female → 42%, Others → 2%
Handedness	Right → 93%; Left → 6%; Ambidextrous → 1%
Countries	India → 47%; USA → 37%; UK → 9%; Others → 7%
Typing	Both hands → 70%; Right hand → 26%; Left hand → 4%
Age	[25-34] → 54%; [18-24] → 22%; [35-44] → 23%; [45-54] → 1%

even "caught" several participants who started typing without manually disabling these features in their settings. At least three participants said during the study that "the prediction here works better than on my phone,", likely because they were using an older phone. In this pilot, 44% of our participants used Android version 9, 23% version 8 and the remaining 33% version 7 or below.

For the last pilot study where all features were enabled, we recruited 89 participants through Amazon Mechanical Turk (MTurk). Relative to what we observed in the previous pilots, some participants adjusted their behaviors and the rate of incorrect submissions, i.e., where undesirable behaviours were observed, dropped to 22%, which corresponds to 19 participants.

5 EXPERIMENT

We used our web-based system to run the main, crowd-sourced study to analyze keyboard usage and user behaviors with autocorrection and prediction. We recruited 170 participants via Amazon Mechanical Turk (MTurk). Their task was to copy 30 phrases randomly chosen from our phrase set. Participants also answered short demographic questions before the main task. The average time to complete the experiment was 9 min (SD 33 sec). We paid $1 for each Human Intelligence Task (HIT).

5.1 Restrictions on Participation

To increase the validity of the data, we used several strategies. Following advice from previous work, we recruited people with higher than 95% approved HITs [20]. To reduce the chances that our task had been completed by bots, we used reCAPTCHAs.

Each participant received a unique code based also on the time stamp when they started the experiment, which was only shown to the user after completing the task. We then paid all participants who provided us with a correct code and who had followed the instructions, i.e., who had turned off their own autocorrection and prediction system (and thus had not triggered the corresponding error message).

5.2 Results

We removed all incomplete, inaccurate, or corrupted data. Here, we also excluded data from participants who experienced technical problems, e.g., caused by an unreliable internet connection, or cases where participants got distracted during a sentence, i.e., phrases where we observed pauses that resulted in an unusually low typing speed, lower than 10 WPM within a phrase.

5.2.1 Demographic. We initially recruited 170 participants. After removing participants with incorrect codes and those who did not follow the instructions, data for 150 participants remained for analysis. Table 1 summarizes the demographic background of these participants. According to our logs, 8% of our participants used Android version 10, 46% used version 9, 31% used version 8, and the remaining 15% version 7 or below.

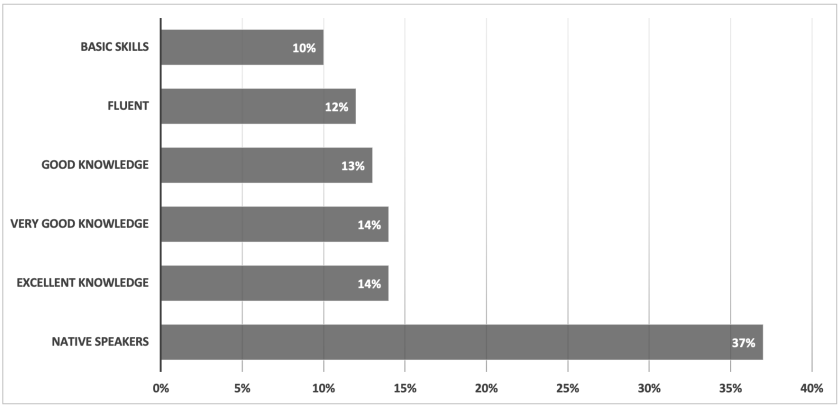


Fig. 2. Distribution of English proficiency.

After they completed our study, we asked participants about their text entry behavior during the task. A majority, 60%, indicated that they typed as fast as possible (which might have compromised their accuracy), while the remaining 40% reported that they were as careful as possible (which might have compromised their speed).

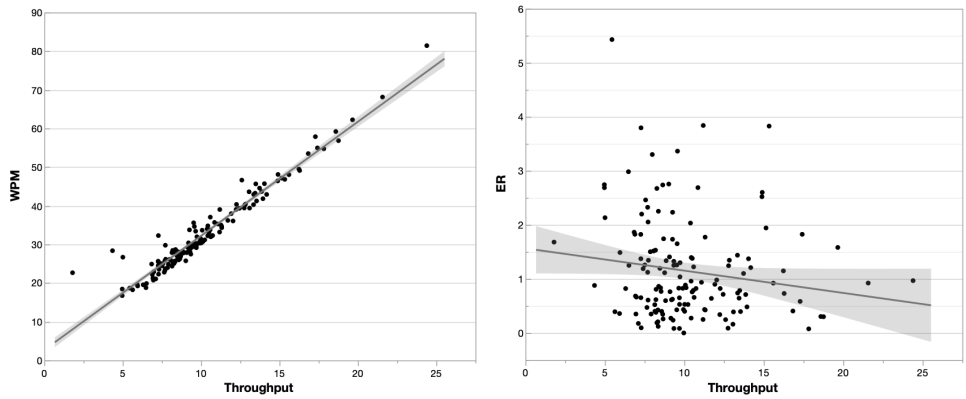
We asked how often they used letters from the Latin alphabet, e.g., the modern English alphabet, in their mobile communication, and provided a corresponding image for additional clarification. 28% reported using it constantly during the day (at least once every 10 minutes), 23% more than once an hour, 24% more than once per day, 17% a few times per week, and 8% rarely (less than once a week).

Even though our task does not require high English proficiency, we asked participants to rate their English proficiency on a scale: no knowledge, basic communication skills, good knowledge, very good knowledge, excellent knowledge/highly proficient, near native/fluent, and native speaker. Figure 2 shows the distribution of the reported English proficiency. For objective analysis, we created an English grammar quiz using test questions from http://iteslj.org, by selecting six question: two easy, two medium, and two hard for our quiz. We then defined the "overall success rate" as the percentage of correct answers that participants achieved in that quiz. Results show that the overall success rate for the English proficiency quiz was 80.5% (SD=23.4), which corresponds to a reasonably high English proficiency.

5.2.2 Dataset Overview. Overall, through our monitoring of text field content changes, we logged 207,962 actions for all 150 participants, as well as 1,238 autocorrection events generated by our implementation, for a total of 209,200 events. Of these, 2,956 (1%) were user interactions with our custom prediction panel, 769 (.4%) cursor movements, and 13,870 (7%) backspaces.

Overall, participants inputted 4,500 phrases, of which 99% were inputted in portrait mode. The average number of input actions per word was 5.35 (SD=1.32), including picks from the prediction panel. The average number of backspaces per word was .34 (SD=.54). The average number of keystrokes per character (KSPC) was 1.14 (SD=0.23).

5.2.3 Typing Speed and Accuracy. We measured typing speed through the words per minute (WPM) metric and accuracy through the error rate (ER) of the submitted text. ER was computed through the minimum string distance error rate (MSD ER) [5]. The average speed of participants was 32.96 WPM (SD=13.37). The average MSD error rate across the whole dataset was 1.02% (SD=.48).



(a) (b)

Fig. 3. Regression analysis of a) WPM, and b) ER, across the throughput.

We also measured the verification time, i.e., the time spent to review the phrase, to be 1.25 seconds (SD=1.71) per phrase, measured from the last keystroke until the participant progressed to the next phrase [1, 4].

Recently, a throughput measure for text entry had been introduced, which was designed to reflect the efficiency of a text entry method [61]. It is defined as the amount of information transferred via a text entry method per unit time. We calculated the throughput using Zhang et al.'s method[3] [61]. The average throughput across the whole dataset was 10.1 bits/second (SD=3.45) per phrase. Then we performed a regression analysis and found a strong correlation between throughput and WPM (r=0.94), but no correlation with error rate (r=.024), see Figure 3.

5.2.4 Prediction Panel Use. The target word appeared 94% (SD=12) of the time in the prediction panel. The total amount of words picked from the prediction panel was 2,956. Users entered on average 2.48 (SD=6.06) characters of a word before a prediction pick event. The average number of words per phrase picked from the prediction panel was 0.66 (SD=1.69). The participants picked the word in the center of prediction panel (the most probable one) 63% of the time, the word on the right (the second most probable one) in 23% of all cases, and the left one (the third most probable one or the originally inputted word) 14% of the time (Figure 1). In total, the number of predicted characters was 15,382, i.e., participants saved an average of 3.43 (SD=8.01) characters per phrase by using the prediction. The average time participants spent between the last keystroke and picking a word from the prediction panel was .97 seconds (SD=1.06). The overall average time to type a word was 4.98 (SD=2.10) seconds, but the average time spent for words picked from the prediction panel was longer, 7.07 seconds (SD=2.36), for a difference of 2.09 seconds (SD=1.87). Participants used the prediction mostly for slightly longer words, as the average length of predicted words was 7 characters (SD=2.34). To examine word complexity objectively, we used the "count words worth" website [4] and found that the complexity of words picked from the prediction panel was average in terms of syllables and readability. The average number of syllables for the predicted words was 1.75 (SD=.76), which matches the average for all English words. According to the Flesch–Kincaid

[3] https://github.com/DrustZ/Throughputrepository
[4] https://countwordsworth.com

method [46] the readability of the words was 81.85, which corresponds to easy-to-read words for conversational English use.

5.2.5 Autocorrection. Overall, we recorded 1,238 occurrences of autocorrection. The average number of autocorrections per phrase was .28 (SD=.62), which corresponds to 3% (SD=.07) of the words in the transcribed text. The most frequent mistake that autocorrect addressed were misspellings of the word "you". Participants usually typed it missing one letter ("yo", "yu") or using incorrect characters like "ypu". The second most frequently corrected word was "the", where users often typed "tge" or "tye", most likely due to finger touches on an adjacent letter. Examples of other frequent mistakes are "calander, probleme, shoud, hav". Of all autocorrections, 93% (M=.26, SD=.56) were successful, but 7% (M=.02, SD=.21) were wrong. For wrong autocorrections, the string distance between the autocorrected and presented word was on average 1.85 characters (SD=1.55) but the average distance between the typed and presented word was only 1.54 characters (SD=1.53).

5.2.6 Cost of Error Correction. The average number of words with writing mistakes during typing was .82 (SD=1.63) per phrase, however the average number of words with mistakes after submitting the phrase was only .30 (SD=.92) per phrase. This means that most of the time users went back and fixed their mistakes. In the following, we define the correction time to be the time difference between when the user starting an editing episode, using either backspace or a cursor movement, until the time user entered new text.

Less than half, 42%, of the inputted phrases had one or more error correction events. Among these phrases, the average time participants took to correct their errors per phrase (excluding autocorrections or word predictions) was 2.72 (SD=2.09) seconds.

The average time to edit a wrong autocorrection was 5.50 (SD=6.36) seconds. For wrong autocorrections, the presented word is typically closer to the original word that participants typed in 52% of the instances. The remaining instances are either closer to the wrong autocorrection, in 21% of cases, or equally close to both 27%. Most of these cases concerned names, of people or companies, such as Socal, Dowd, Diane, Whitt, Lara, Gallup, Omaha, Stan, and Doyle. In addition, participants entered unrecognized words due to spelling or typing mistakes, such as "sholid", which was autocorrected to "solid". Yet, the target sentence contained the word "should". The misprediction was sometimes due to the prediction adding additional characters, for example "ma" being autocorrected to "man". Yet, in this case the correct word was supposed to be "am". In 36% of these instances, users did not notice the changes in the text or did not care to fix the wrong autocorrection. Yet, in 30% of the cases, they noticed the issue immediately and corrected it through an undo of the autocorrection (through backspacing on the autocorrected word). An additional 8% of errors were corrected by moving the cursor during the (final) verification time and 4% directly after autocorrection by adding or editing one letter. 12% were corrected through using backspace after entering one more word, 8% immediately after the autocorrection occurred, and only 2% through backspaces after typing more than one word.

In 48 instances, participants picked a word from the prediction panel and then went back to change or add to this word. The average time participants took to edit a word previously chosen from the prediction panel, 8.03 (SD=12.45) seconds, was long and varied a lot. In (only) four of these instances participants adjusted the word-ending of a "close enough" prediction, which took on average only .69 seconds (SD=.27).

We compared phrases with error correction events and those without them through an independent-samples t-test. We found the error correction events significantly decreased speed, $t(4488)=24.38$, $p<.0005$, with an absolute difference of 2.49 seconds.

Similarly, we compared phrases with and without word prediction events through an independent-samples t-test. We found that the use of word prediction significantly decreased the text entry

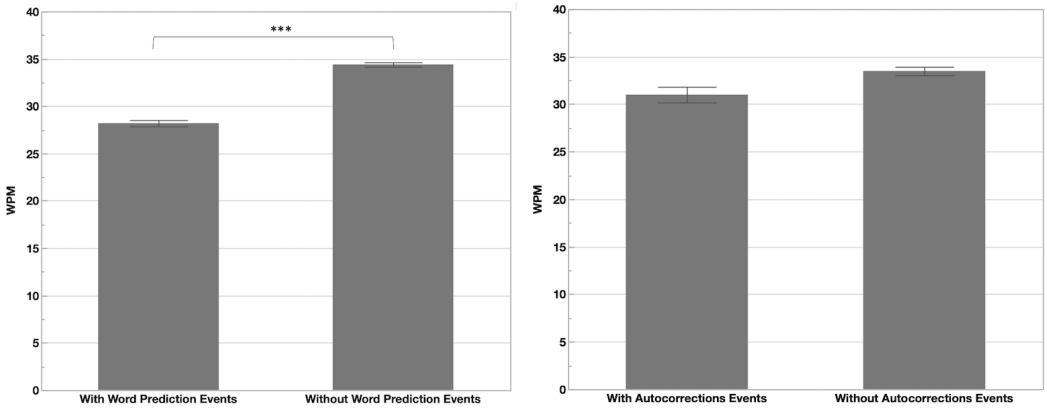


Fig. 4. Comparison of text input speed for phrases with and without a) word prediction and b) autocorrection events.

speed, t(4488)=13.99, p<.0005, see Figure 4. This indicates that word prediction slows down text entry.

We also separated phrases with and without autocorrections events and conducted again an independent-samples t-test. We found that autocorrection did not have a statistically significant effect on text entry speed, t(4488)=5.15, p=.61, see Figure 4.

6 DISCUSSION

Recently, several mobile typing studies had been conducted outside the lab to observe more realistic user behaviors, e.g., [9, 37]. Similarly, we also employed an online crowd-sourcing approach to reach a larger group of more diverse participants. We used a transcription typing to objectively measure participants' typing speed and error rate without introducing the overhead associated with creating compositions [54].

To assess the correctness of predictions and auto-corrections we used a combination of our own predictive system and a controlled transcription task. We also chose this approach due to potential privacy concerns, as using participants' own predictions might reveal personal details. Yet, as mentioned above, our word prediction feature matched the one in Android 9 fairly closely. The target word appeared 94% (SD=12) of the time in the prediction panel (see section 4.2).

One of the main outcomes of our work is that we identified that choosing words from the prediction panel can compromise text entry speed and too frequent use of the prediction panel can decrease text entry speed. Our results show that typing a word is faster than picking that word from prediction panel, with a difference of an average of 2.09 seconds per word. Based on our data, participants took on average of .97 seconds before choosing a word from the prediction panel, which includes the (non-negligible) overhead of switching attention, as explored in previous work, e.g., [13, 51]. A potential reason behind this relatively long time is that users need to read the three entries within the prediction panel to make a choice which one to select, and then carry out that selection.

Almost two-thirds of the time (63%) users chose the most probable word in the center of the prediction panel (Figure 1). That fact could motivate new interactions around the most probable

choice that makes this word stand out more and/or further decreases finger travel time between the keys and the prediction panel, e.g., [39]. The costs associated with using autocorrection and word prediction mechanisms could also justify research to re-imagine text entry interfaces more radically, e.g., by leveraging information beyond the keystrokes, such as pressure [56] or gestures [2], allowing new forms of interactions with the entered text [60], or by moving completely away from keyboard-centric input, e.g., by using (vocal or silent) speech, handwriting, or airwriting.

Our 1,238 recorded autocorrections correspond to .6% of all events (see section 5.2.2), which matches the .7% reported by Buschek et al. [9] in their "in the wild" study reasonably well. Our results also identify that about half the time functionality to undo an autocorrection is beneficial when the autocorrection method failed, since about 52% of the time the entered word is closer to the presented/target word. These words became unrecognizable to the autocorrection method due to a combination of instances where the user typed an adjacent key and also made spelling mistakes. The rest of the time the distance is either equal or closer to the autocorrected word.

In terms of average typing speeds, our result (32.96 WPM) matches comparable work, both "in the wild", e.g., 32.1 WPM [9], and controlled lab studies, e.g., 31.1 WPM [17]. A recent study reported a somewhat higher speed of 36.17 WPM [37], which might be understandable since this study used a speed-test game. To our knowledge, we are the first large-scale study to use the new text entry throughput metric [61]. We performed a regression analysis and found a strong correlation between the new throughput metric and WPM (r=0.94) but not with error rate (r=.024), see Figure 3. In contrast to what the work that introduced text entry throughput claims [61], this metric failed to adequately account for varying error rates. That makes us believe that the throughput metric calculation based on approximations [61], which we used, is not yet method-independent. That means that for each application or keyboard layout, one would need to calculate the exact probability for each error instance to get the full benefit of this throughput measure.

To get externally valid insights from the study, we chose crowd-sourcing to examine the effect within a larger population. Since our participants were representative of a reasonably wide range of English proficiency, demographics, geographic distribution, and typing behaviors (in terms of speed/accuracy emphasis), the results of our crowd-sourced study can thus potentially generalize to an even larger audience. Our typing task setting imposes a more controlled environment relative to "in the wild" studies. We asked participants to use their own keyboard because we wanted to avoid the overhead associated with learning a potentially new layout. Most of our participants from around the world had good English-language knowledge. Due to regional linguistic diversity, many countries, e.g., India, use English for communication, both internationally and domestically. We verified this objectively through the English test and subjectively with the self-reports. More than a third of our population indicated that they are native speakers. Previous studies, e.g, [37], targeted experienced participants, mostly from the USA, which does not necessarily correspond well to a larger population. Due to globalization, many non-native English speakers still communicate in English. Thus, results for native speakers may not be representative of the majority of global users.

East Asian languages such as Chinese and Japanese use ideograms in writing. In such languages every morpheme (the smallest unit carrying meaning) corresponds to a distinct character, thus the number of characters in the language is large. In both Chinese and Japanese, the standard character set includes more than 6,000 characters [49]. When computers were introduced, the greatest challenge for users of these languages was how to enter such a large number of characters - with keyboards designed for European languages[30]. Most Chinese or Japanese users enter text using predictive methods [49]. The prediction systems for East Asian languages rely either on a phonetically- or a shape-based approach [29, 35]. Users enter a sequence of phonograms or Latin alphabetic transcriptions, then the system sorts the candidates in a relevant order, and displays them for the user, who chooses their preferred target among the candidates. Compared to English,

such text entry methods use fundamentally different methods to process the language and generate predictions. Thus, we believe that our results are not directly generalizable to such languages.

Current touchscreen keyboards depend on language models and dictionaries to provide word predictions, to correct touch-location errors, to autocomplete words, and to predict what the user will type next [15]. A language model can be word-based, sentence-based, or both. It can adapt to the user by utilizing the users' writing history. Adaptive models are able to better reflect a user's usage of language. The best prediction methods are the ones that combine both word-based and sentence-based prediction and the ones that utilize the users' written history [23, 58]. That said, because these innovations are only available with the newest operating system version(s), most text input systems that are used today on a daily basis around the world use simpler word-based prediction and autocorrection methods.

Thus, a limitation of our study is that the latest mobile devices offer more advanced algorithms relative to what we used in this experiment. However, our international participant pool exhibited a mixture of Android versions. Thus, participants were, on average, using an older version of Android's autocorrect and prediction methods, which do not support the features that are reported in the most recent text entry literature. As mentioned above, the system we used here matches the behaviour of a slightly out-of-date Android device quite well. We re-emphasize here that the main focus of our study was to identify user behaviors around autocorrect and prediction errors, regardless of the specifics of the predictive algorithm. Still, in our pilot studies, where we asked participants directly about this, all participants replied that they had not noticed differences with the prediction and autocorrect methods that they were familiar with. Another potential limitation of our work is that our data collection system is web-based, which might have delayed some system responses slightly, e.g., when skipping to the next phrase and logging the episode string into the corresponding log file. Yet, according to our observations this did not introduce any noticeable effects.

7 CONCLUSION

We investigated behaviors that people exhibit around predictions and autocorrections in text entry. We collected a comprehensive dataset using a novel data collection system that observes user behaviors through a crowd-sourcing platform. While our results identify that autocorrection does not have an effect on text entry speed, we found that choosing words from the prediction panel can compromise text entry speed and too frequent use of the prediction panel can even decrease text entry speed. Participants paused on average about one second before choosing a word from the prediction panel, which is likely due to the overhead of an attention switch. Our results also identify that functionality to undo autocorrection, i.e., to revert back to the original input, is beneficial in about half of all wrong autocorrect episodes, since users entered unrecognizable words, often through tapping on adjacent keys in combination with spelling mistakes.

7.1 Future Work

In future studies we are considering to investigate user behaviors with autocorrection and predictive systems using a sentence-based approach and to cover other languages besides English. We are also planning to investigate new interaction methods for the most probable word to minimize the overhead for switching attention.

8 ACKNOWLEDGMENTS

We thank the participants in our studies. The work was funded by a generous scholarship from the Saudi Arabian Cultural Bureau in Ottawa and King Saud University, and partially supported by

NSERC and DFG grant no. 316930318 "Detection of Interaction Competencies and Obstacles" to whom we are grateful.

REFERENCES

[1] Ohoud Alharbi, Ahmed Sabbir Arif, Wolfgang Stuerzlinger, Mark D. Dunlop, and Andreas Komninos. 2019. WiseType: A tablet keyboard with color-coded visualization and various editing options for error correction. In *Proceedings - Graphics Interface*. https://doi.org/10.20380/GI2019.04

[2] Jessalyn Alvina, Joseph Malloch, and Wendy E. Mackay. 2016. Expressive keyboards: Enriching gesture-typing on mobile devices. In *UIST 2016 - Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. https://doi.org/10.1145/2984511.2984560

[3] Denis Anson, Penni Moist, Mary Przywara, Heather Wells, Heather Saylor, and Hantz Maxime. 2006. The effects of word completion and word prediction on typing rates using on-screen keyboards. *Assistive Technology* (2006). https://doi.org/10.1080/10400435.2006.10131913

[4] Ahmed Sabbir Arif, Sunjun Kim, Wolfgang Stuerzlinger, Geehyuk Lee, and Ali Mazalek. 2016. Evaluation of a smart-restorable backspace technique to facilitate text entry error correction. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (2016)* (2016), 5151–5162. https://doi.org/10.1145/2858036.2858407

[5] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *Proceedings of the IEEE Toronto International Conference - Science and Technology for Humanity - TIC-STH '09*. IEEE, 100–105. https://doi.org/10.1109/TIC-STH.2009.5444533

[6] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2010. Predicting the cost of error correction in character-based text entry technologies. In *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*. https://doi.org/10.1145/1753326.1753329

[7] Kenneth C. Arnold, Krzysztof Z. Gajos, and Adam T. Kalai. 2016. On Suggesting Phrases vs. Predicting Words for Mobile Text Composition. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16* (2016), 603–608. https://doi.org/10.1145/2984511.2984584

[8] Xiaojun Bi and Shumin Zhai. 2016. IJQwerty: What Difference Does One Key Change Make? Gesture Typing Keyboard Optimization Bounded by One Key Position Change from Qwerty. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16* (2016). https://doi.org/10.1145/2858036.2858421

[9] Daniel Buschek, Benjamin Bisinger, and Florian Alt. 2018. ResearchIME: A Mobile Keyboard Application for Studying Free Typing Behaviour in the Wild. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18* (2018). https://doi.org/10.1145/3173574.3173829

[10] Daniel Buschek, Julia Kinshofer, and Florian Alt. 2018. A Comparative Evaluation of Spatial Targeting Behaviour Patterns for Finger and Stylus Tapping on Mobile Touchscreen Devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (2018). https://doi.org/10.1145/3161160

[11] Tanzeem Choudhury, Gaetano Borriello, Sunny Consolvo, Dirk Haehnel, Beverly Harrison, Bruce Hemingway, Jeffrey Hightower, Predrag "Pedja" Klasnja, Karl Koscher, Anthony LaMarca, James A. Landay, Louis LeGrand, Jonathan Lester, Ali Rahimi, Adam Rea, and Danny Wyatt. 2008. The Mobile Sensing Platform: An Embedded Activity Recognition System. *IEEE Pervasive Computing* (2008). https://doi.org/10.1109/MPRV.2008.39

[12] Vivek Dhakal, Anna Maria Feit, Per Ola Kristensson, and Antti Oulasvirta. 2018. Observations on typing from 136 million keystrokes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18 (2018)*. https://doi.org/10.1145/3173574.3174220

[13] Mark D. Dunlop and Andrew Crossan. 2000. Predictive text entry methods for mobile phones. *Personal Technologies* (2000), 134–143. https://doi.org/10.1007/BF01324120

[14] Abigail Evans and Jacob O. Wobbrock. 2012. Taming Wild Behavior: The Input Observer for Obtaining Text Entry and Mouse Pointing Measures from Everyday Computer Use. In *Proceedings of the 2012 CHI Conference on Human Factors in Computing Systems - CHI '12 (2012)*. https://doi.org/10.1145/2207676.2208338

[15] Andrew Fowler, Kurt Partridge, Ciprian Chelba, Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2015. Effects of Language Modeling and its Personalization on Touchscreen Typing Performance. *Proceedings of the ACM CHI'15 Conference on Human Factors in Computing Systems* 1 (2015), 649–658. https://doi.org/10.1145/2702123.2702503

[16] Jon Froehlich, Mike Y Chen, Sunny Consolvo, Beverly Harrison, and James A Landay. 2007. MyExperience: a system for in situ tracing and capturing of user feedback on mobile phones. In *The 5th International Conference on Mobile Systems, Applications, and Services*. https://doi.org/10.1145/1247660.1247670

[17] Mayank Goel, Leah Findlater, and Jacob O. Wobbrock. 2012. WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. https://doi.org/10.1145/2207676.2208662

[18] Niels Henze, Enrico Rukzio, and Susanne Boll. 2012. Observational and experimental investigation of typing behaviour using virtual keyboards on mobile devices. In *InProceedings of the 2012 CHI Conference on Human Factors in Computing Systems - CHI '12 (2012)*. https://doi.org/10.1145/2207676.2208658

[19] Cl James and Km Reischel. 2001. Text input for mobile devices: comparing model prediction to actual performance. *Proceedings of the 2001 CHI Conference on Human Factors in Computing Systems - CHI '01 (2001)* (2001), 365–371. https://doi.org/10.1145/365024.365300

[20] Anthony Jameson. 2007. Adaptive Interfaces and Agents. In *The human-computer interaction handbook 2007*. CRC Press, 459–484.

[21] Anthony Jameson and Per Ola Kristensson. 2017. Understanding and supporting modality choices. In *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations - Volume 1*. ACM, 201–238. https://doi.org/10.1145/3015783.3015790

[22] Alex Jansen, Findlater Leah, and Jacob O. Wobbrock. 2011. From the lab to the world: Lessons from extending a pointing technique for real-world use. *In CHI'11 Extended Abstracts on Human Factors in Computing Systems* (2011).

[23] Shaoxiong Ji, Shirui Pan, Guodong Long, Xue Li, Jing Jiang, and Zi Huang. 2019. Learning Private Neural Language Modeling with Attentive Aggregation. In *Proceedings of the International Joint Conference on Neural Networks*. https://doi.org/10.1109/IJCNN.2019.8852464

[24] Jussi P.P. Jokinen, Sayan Sarcar, Antti Oulasvirta, Chaklam Silpasuwanchai, Zhenxin Wang, and Xiangshi Ren. 2017. Modelling learning of new keyboard layouts. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '16 (2017)*. https://doi.org/10.1145/3025453.3025580

[25] Heidi Horstmann Koester and Simon P. Levine. 1994. Modeling the Speed of Text Entry with a Word Prediction Interface. *IEEE Transactions on Rehabilitation Engineering* 2, 3 (1994), 177–187. https://doi.org/10.1109/86.331567

[26] Heidi Horstmann Koester and Simon P. Levine. 1996. Effect of a word prediction feature on user performance. *AAC: Augmentative and Alternative Communication* (1996). https://doi.org/10.1080/07434619612331277608

[27] Per Ola Kristensson and Keith Vertanen. 2012. Performance comparisons of phrase sets and presentation styles for text entry evaluations. In *International Conference on Intelligent User Interfaces, Proceedings IUI*. https://doi.org/10.1145/2166966.2166972

[28] Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady* 10, 8 (1966), 707–710.

[29] Ying Liu, Kai Ding, and Ning Liu. 2009. Immediate user performances with touch Chinese text entry solutions on handheld devices. In *ACM International Conference Proceeding Series*.

[30] Ying Liu and Kari Jouko Räihä. 2010. Predicting Chinese text entry speeds on mobile phones. In *Conference on Human Factors in Computing Systems - Proceedings*. https://doi.org/10.1145/1753326.1753657

[31] I. Scott MacKenzie and R. William Soukoreff. 2002. Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction* (2002). https://doi.org/10.1207/S15327051HCI172

[32] I. Scott MacKenzie and Shawn X. Zhang. 1999. The design and evaluation of a high-performance soft keyboard. In *Proceedings of the SIGCHI conference on Human factors in computing systems the CHI is the limit - CHI '99*. ACM Press, New York, 25–31. https://doi.org/10.1145/302979.302983

[33] Mishaal Rahman. 2020. Android Version Distribution statistics will now only be available in Android Studio. https://www.xda-developers.com/android-version-distribution-statistics-android-studio/

[34] Emma Nicol, Andreas Komninos, and Mark D Dunlop. 2016. A Participatory Design and Formal Study Investigation into Mobile Text Entry for Older Adults. *International Journal of Mobile Human Computer Interaction* 8, 2 (2016), 20–46. https://doi.org/10.4018/IJMHCI.2016040102.oa

[35] Jianwei Niu, Yang Liu, Jialiu Lin, Like Zhu, and Kongqiao Wang. 2014. Stroke++: A new Chinese input method for touch screen mobile phones. *International Journal of Human Computer Studies* (2014). https://doi.org/10.1016/j.ijhcs.2014.01.001

[36] Sharon Oviatt, Bjorn Schuller, Philip R. Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Kruger. 2017. *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations - Volume 1*. 633 pages. https://doi.org/10.1145/3015783

[37] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do people type on mobile devices? Observations from a study with 37,000 volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI 2019*. https://doi.org/10.1145/3338286.3340120

[38] Felix Putze, Tilman Ihrig, Tanja Schultz, and Wolfgang Stuerzlinger. 2020. Platform for Studying Self-Repairing Auto-Corrections in Mobile Text Entry based on Brain Activity, Gaze, and Context. https://doi.org/10.1145/3313831.3376815

[39] Felix Putze, Maik Schünemann, Tanja Schultz, and Wolfgang Stuerzlinger. 2017. Automatic classification of auto-correction errors in predictive text entry based on EEG and context information. In *ICMI 2017 - Proceedings of the 19th ACM International Conference on Multimodal Interaction*. https://doi.org/10.1145/3136755.3136784

[40] Philip Quinn and Andy Cockburn. 2018. Loss Aversion and Preferences in Interaction. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18* (2018). https://doi.org/10.1080/07370024.2018.1433040

[41] Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16* (2016), 83–88. https://doi.org/10.1145/2858036.2858305

[42] Shyam Reyal, Shumin Zhai, and Per Ola Kristensson. 2015. Performance and User Experience of Touchscreen and Gesture Keyboards in a Lab Setting and in the Wild. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* (2015), 679–688. https://doi.org/10.1145/2702123.2702597

[43] Yvonne Rogers. 2011. Interaction design gone wild: striving for wild theory. *interactions* (2011). https://doi.org/10.1145/1978822.1978834

[44] Richard Schlögl and Thomas Grechenig. 2019. Hyper Typer : A Serious Game for Measuring Mobile Text Entry Performance in the Wild menu. (2019), 1–6.

[45] Klaus U. Schulz and Stoyan Mihov. 2003. Fast string correction with Levenshtein automata. *International Journal on Document Analysis and Recognition* (2003). https://doi.org/10.1007/s10032-002-0082-8

[46] Luo Si and Jamie Callan. 2001. A statistical model for scientific readability. https://doi.org/10.1145/502585.502695

[47] Shyamli Sindhwani, Christof Lutteroth, and Gerald Weber. 2019. ReType: Quick Text Editing with Keyboard and Gaze. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19* (2019), 1–13. https://doi.org/10.1145/3290605.3300433

[48] R. William Soukoreff and I. Scott MacKenzie. 2003. Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. In *Proceedings of the conference on Human factors in computing systems - CHI '03*, Fabio Paternò (Ed.). ACM Press, New York, 113–120. https://doi.org/10.1145/642611.642632

[49] Kumiko Tanaka-Ishii, Jin Dong Kim, and Ming Zhou. 2007. Text Entry in East Asian Languages. In *Text Entry Systems*. https://doi.org/10.1016/B978-012373591-1/50011-5

[50] Pin Shen Teh, Andrew Beng Jin Teoh, and Shigang Yue. 2013. A Survey of Keystroke Dynamics Biometrics. *The Scientific World Journal* (2013). https://doi.org/10.1155/2013/408280

[51] Keith Trnka, John Mccaw, Debra Yarrington, Kathleen F. Mccoy, and Christopher Pennington. 2009. User interaction with word prediction:The effects of prediction quality. *ACM Transactions on Accessible Computing* (2009). https://doi.org/10.1145/1497302.1497307

[52] Keith Vertanen, Justin Emge, Haythem Memmi, and Per Ola Kristensson. 2014. Text blaster. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. https://doi.org/10.1145/2559206.2574802

[53] Keith Vertanen and Per Ola Kristensson. 2011. A versatile dataset for text entry evaluations based on genuine mobile emails. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI '11*. ACM Press, New York, 295. https://doi.org/10.1145/2037373.2037418

[54] Keith Vertanen and Per Ola Kristensson. 2014. Complementing text entry evaluations with a composition task. *ACM Transactions on Computer-Human Interaction* 21, 2 (2014), 1–33. https://doi.org/10.1145/2555691

[55] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. 2015. VelociTap: Investigating Fast Mobile Text Entry using Sentence-Based Decoding of Touchscreen Keyboard Input. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* (2015), 659–668. https://doi.org/10.1145/2702123.2702135

[56] Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain text entry on mobile devices. In *Proceedings of the 2014 CHI Conference on Human Factors in Computing Systems - CHI '14 (2014)*. https://doi.org/10.1145/2556288.2557412

[57] Jacob O. Wobbrock. 2007. Measures of Text Entry Performance. In *Text Entry Systems*. https://doi.org/10.1016/B978-012373591-1/50003-6

[58] Xing Wu, Zhaowang Liang, and Jianjia Wang. 2020. Fedmed: A federated learning framework for language modeling. *Sensors* (2020). https://doi.org/10.3390/s20144048

[59] Xin Yi, Chun Yu, Weijie Xu, Xiaojun Bi, and Yuanchun Shi. 2015. ATK: Enabling Ten-Finger Freehand Typing in Air Based on 3D Hand Tracking Data. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15* (2015). https://doi.org/10.1145/2807442.2807510

[60] Mingrui Zhang, He Wen, and Jacob O. Wobbrock. 2019. Type, then correct: Intelligent text correction techniques for mobile text entry using neural networks. In *UIST 2019 - Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. https://doi.org/10.1145/3332165.3347924

[61] Mingrui Zhang, Shumin Zhai, and Jacob O. Wobbroc. 2019. Text Entry Throughput. *2019 CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019)*, (2019), 1–13.

Received July 2020; revised August 2020; accepted September 2020