WatchWriter: Tap and Gesture Typing on a Smartwatch Miniature Keyboard with Statistical Decoding

Mitchell Gordon *
University of Rochester
m.gordon@rochester.edu

Tom Ouyang
Google Inc.
ouyang@google.com

Shumin Zhai
Google Inc.
zhai@acm.org

ABSTRACT

We present WatchWriter, a finger operated keyboard that supports both touch and gesture typing with statistical decoding on a smartwatch. Just like on modern smartphones, users type one letter per tap or one word per gesture stroke on WatchWriter but in a much smaller spatial scale. WatchWriter demonstrates that human motor control adaptability, coupled with modern statistical decoding and error correction technologies developed for smartphones, can enable a surprisingly effective typing performance despite the small watch size. In a user performance experiment entirely run on a smartwatch, 36 participants reached a speed of 22–24 WPM with near zero error rate.

Author Keywords

WatchWriter; mobile input; text entry; soft keyboard; smartwatch

ACM Classification Keywords

H.5.2. User Interfaces: Input devices and strategies

INTRODUCTION

The smartwatch is emerging as another major category of personal computing devices after the desktop PC, laptops, smart phones, and tablets. One limitation of the current generation of smartwatches is their lack of manual text input. Their inability to send or respond to messages in a quick and private manner is a major user experience bottleneck to a richer and fuller information and communication experience.

The primary challenge to effective manual text input on a smartwatch is the relatively small display size. A normal watchface's width measures two to three human fingers. For a regular Qwerty keyboard whose top row has 10 letter keys, the landing finger may cover three or more key spaces.

Although the wide finger and narrow watch face combination seems like a daunting combination, two factors may come to the rescue. One is on the human side. Human dexterity and adaptability can be remarkable. Humans can generally make more precise movements by slowing down, although such a trade-off becomes highly nonlinear when it approaches the "absolute precision range" of the finger [3]. The second factor is the machine intelligence capabilities developed for smartphones in past decade. Modern touch screen keyboards use statistical inference from language and spatial modeling to decode the user's messy, ambiguous, or erroneous input into the most likely text intended by the user. They also typically offer alternative and next word predictions. For example, the raw key-level error rate on a typical smartphone keyboard hovers around 9% (varying slightly depending on hand posture [2]), which translates to a word-level error rate of about 40%. That smartphone keyboards work at all relies on the fact that most of these spatial errors are corrected automatically, often without the user even noticing.

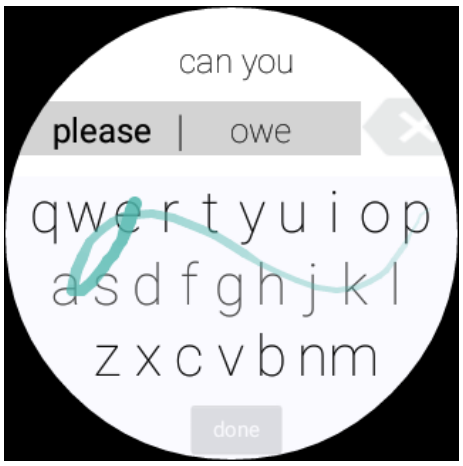

Figure 1. WatchWriter supports both predictive tap and gesture typing. It demonstrated that single-step tap and gesture typing can be quite effective even on a small watch form factor. In this image, a user is finishing a gesture for the word "please."

By porting (the language and spatial models and decoder) and reimplementing (UI) technologies and modules of touchscreen keyboards we developed over many years for Android smartphone keyboards (in particular Google Keyboard), we built a smartwatch keyboard prototype (WatchWriter) for the Android Wear operating system (Figure 1). WatchWriter is capable of both touch (tap) and gesture (or surface stroke, also known as shape writing or swipe) typing. We were very surprised by our own initial and informal test experience with the prototype. This paper reports more formal empirical results obtained in a user study of WatchWriter.

*This work was done while Mitchell Gordon was an intern at Google Inc.

CHI'16, May 07-12, 2016, San Jose, CA, USA
ACM 978-1-4503-3362-7/16/05.
http://dx.doi.org/10.1145/2858036.2858242

RELATED WORK

Multi-step Smartwatch Keyboards

In order to overcome the precision challenge of pinpointing individual keys, researchers have proposed and researched various interaction methods that break up each key selection into a multi-step operation. ZoomBoard [10] is a Qwerty keyboard that works by first displaying a full keyboard, then requires a tap to visually zoom in on a smaller area of the keyboard. To select a key, users tap again on the new smaller set of visually larger keys. ZoomBoard recorded 9.3 WPM (words per minute) at 2.15 KSPC (keystrokes per character) and 1.4% CER (character error rate). SplitBoard [5] took the Qwerty keyboard and displayed half of it on the screen at a time, requiring a separate gesture to switch between halves. SplitBoard recorded 16.3 WPM, though no evaluation of SplitBoard has reported KSPC or CER. Both ZoomBoard and SplitBoard tested "dumb" Qwerty keyboards as a baseline, and reported 4.5 WPM and 13.7 WPM, respectively. Chen et. al. created SwipeBoard, in which characters are entered with two swipes; the first swipe specifies the region where the character is located, and the second swipe specifies the character within that region [4]. SwipeBoard reported initial performance of 9.1 WPM, with a large increase to 19.6 WPM after 2 hours of practice. SwipeBoard does not report CER, but reports error rates of 4.2% for their first step and 13.3% for their second step. The authors note that an expert-level SwipeBoard user could "chunk" a single character entry into an efficient and eyes-free double swipe, making it a single-step keyboard for expert users.

Neither ZoomBoard nor SwipeBoard evaluated their keyboards on a physical smartwatch, but instead displayed them in small windows on phone or tablet screens without the same watch ergonomic factors such as size or orientation on wrist.

None of the above multi-step research keyboards used statistical decoding for error correction or text prediction. The focus was on methods alternative to single step taps to effectively overcome the size limitation of watches. It is not clear how language model based decoding could be combined with these methods, and whether it could be useful or effective at all.

Single-step Smartwatch Keyboards

To test the power of sentence level decoding for smart keyboards, Vertanen and colleagues [13] tested their tap typing prototype decoder on a watch sized layout on a smartphone (with recognition taking place on a 3.6 GHz 8-core server) as one of the experimental conditions. They showed that participants performance on the watch sized keyboard did not significantly drop from that on a phone sized keyboard (from 40.6 WPM to 34.9 WPM, N.S.) but increased errors from 3.0% to 10.7% CER. The latter CER is similar to that of the raw 9% CER on smartphones without decoding, as reported in [2]. One of the authors of [13] could personally reduce the error rate on their prototype to a practically useful level.

Leiva et al. introduced the ZShift keyboard [8]. Users of ZShift touch a key to select an initial character, and then optionally make slight adjustments with their finger to select surrounding keys. To prevent the user's finger from obscuring their view of the keys, ZShift displays a magnified preview above the keyboard area. ZShift's largest screen size of 32mm, closest to that of a typical smart watch, recorded 9.1 WPM at 1.3 KSPC and 0.9% CER when evaluated on a smartphone. ZShift did not use statistical decoding. Komninos and Dunlop evaluated a clustered alphabetical layout on a smartwatch that did use statistical decoding, and found a WPM of 8.1 [6].

Microsoft has released a smart touch keyboard on their Band fitness device that is much narrower than a typical smartwatch. This product level implementation seemed to be quite accurate at decoding tap typing. Little has been formally reported on the user speed and error performance of such a miniature touch keyboard.

To our knowledge, no gesture typing functions has been previously reported on a smartwatch. Gesture typing tended to be faster than one finger taping, particularly "in the wild" (amid daily activities). The world's first publicly demonstrated gesture keyboard, SHARK [7], showed their keyboard height at slightly larger than an US quarter coin, although SHARK was operated with a stylus on a Windows tablet PC.

Smart Touch Keyboard and Smart Gesture Keyboard

WatchWriter builds on smart touch keyboard and smart gesture keyboard technologies already well established on smartphone platforms. For the latest and perhaps the most comprehensive published smartphone STK and SGK empirical evaluation, see Reyal et al [11].

SYSTEM

We built WatchWriter as an Android Wear keyboard that shares much of the underlying technology with the previously publicly released Google Keyboard on Android phones and tablets. This allows us to test the current smartwatch system's technical capability and the human performance and ergonomics of tap and gesture typing in watch scale and form factor. Among the components shared with Google Keyboard is the statistical decoder, which powers correction and gesture recognition capabilities (processing takes place on the watch itself). It employs a 160k+ word English vocabulary, a large n-gram language model, and spatial models for both touch (tap) and gesture typing input. Please see this paper's video figure for a demonstration of WatchWriter.

Like on many smartphone keyboards, WatchWriter features a suggestion bar that presents the user with two possible recognitions for the word that they are currently composing. The entries in the suggestion bar automatically update upon each key press, and can show either corrections (e.g., from "tje" to "the") or prefix-completions (e.g., from "tomo" to "tomorrow"). In order to commit the current word and move on to the next one, the user taps on one of these suggestions (in gesture typing, the user may also simply perform their next word's gesture and the best suggestion for the current word is automatically committed). These suggestion taps also automatically insert a space after the committed word, eliminating the need for a dedicated spacebar key and saving valuable

screen space. Another advantage of requiring deliberate suggestion taps to commit words is that users are much more aware of what gets inserted into the text view. It removes the well-known danger of having a spacebar tap auto-correct to an unintended word without the user noticing, leading to the types of humorous autocorrect "fails" posted to websites like www.damnyouautocorrect.com. In gesture typing, the choices on the suggestion bar are the keyboard's two best predictions given input and context. In tap typing, the bold suggestion on the left side of the suggestion bar contains the most likely prediction as decided by the decoder, and the right suggestion contains the literal string. If the most likely prediction matches the literal string, then the left side of the suggestion bar displays the second most likely prediction instead.

Upon tapping an item in the suggestion bar, that text is then added to the running output string above, and the suggestion bar is cleared in preparation for a new word.

WatchWriter's backspace key first clears the currently-composing word. Further presses clear words in the output string one at a time. The backspace key always operates at a word-by-word level rather than a character-by-character level. This behavior is typical of gesture typing keyboards because each gesture represents an entire word, and if the suggestions do not contain the desired word, the user must try gesturing the entire word again. We elected to keep this behavior for tap typing as well to encourage users to take advantage of the keyboard's auto-correction and prediction capabilities to compose their word, rather than rely on character-by-character edits on the literal string.

Additional Keyboard Layouts

In addition to a traditional Qwerty layout, WatchWriter also supports three non-Qwerty layouts. We studied these layouts along with the standard Qwerty on a watch form factor. We limit this Note to Qwerty results only. We plan to publish non-Qwerty WatchWriter layout descriptions, motivations, and results when ready in the future. Suffice it to say the three non-Qwerty layouts were not competitive to the familiar Qwerty without learning, even though they may fit the form or precision factors of a watch better.

USER STUDY

Design

The user study consisted of two separate tests, one for gesture typing and one for tap typing. In Test 1, 18 participants gesture typed a fixed set of 21 phrases on four keyboard layouts: Qwerty and three other non-Qwerty layouts. In Test 2, another 18 participants tap typed the same set of 21 phrases on three layouts: Qwerty and two other non-Qwerty layouts. In both tests the order of layouts were balanced by a Latin Square design. As stated earlier, the alternative layouts are of no interest to the current paper. Each participant completed the study in around 45 minutes in total.

The 21 phrases were selected at random from MacKenzie and Soukoreff's list of phrases for evaluating text entry techniques due to their memorability [9]. These phrases consisted of common English-language words. Participants were instructed to correct their entry errors as best they could.

Participants

The 36 participants were employees or interns in the technology industry. Their ages ranged from early 20s to mid 40s, with the average age in the mid 20s. Approximately 1/3 of participants were female. All were fluent in English.

For Test 1 (gesture) we recruited participants with a mix of gesture typing experience: 1/3 were self claimed proficient gesture typing users, 1/3 had some experience, and 1/3 had never used gesture typing before. For users with no gesture typing experience, we had them enter a few short practice phrases using Google Keyboard on a Nexus 5 phone prior to the experiment. No training was provided for Test 2 (tap) as all participants were familiar with tap typing on soft keyboards. All participants received an incentive worth $15 USD.

Procedure

We conducted our study on a LG G Watch R smartwatch running Android Wear, with WatchWriter as an application on the watch. This is a round watch with a 1.3" circular display.

Participants were given the watch and told to wear it on whichever wrist was most comfortable for them. All participants chose to sit with their wrist resting on a table. All participants but one chose to type using their index finger, with the other participant choosing to use their thumb. In order to get accurate timing data, an experiment application ran on the watch and guided users through the test. The application prompted the participant with a short phrase to memorize. Upon pressing "continue" the application showed the participant the keyboard and the participant typed in the phrase, pressing a "done" button when finished. Participants were instructed to correct their errors as best they could, but told that if they saw an error several words back, they did not need to correct it. After pressing "done," the application then prompted the participant with the next phrase. Participants were able to take up to a 60 second break between entering phrases. After the entry task, participants were given a short questionnaire and a chance to provide verbal feedback.

Metrics

We used the following common metrics to measure the user experience of WatchWriter in comparison to other smartwatch and smartphone keyboards.

WPM (words per minute) is a standard metric for reporting typing speed. Because word length varies, in the HCI literature WPM is typically standardized so that each word is made up of 5 characters, including spaces [1]. We follow this.

KSPC (keystrokes per character) reports the number of individual actions performed per character typed [12]. We count a gesture as a single keystroke, meaning that a single keystroke could in the best case be sufficient to input an entire word. We also include tapping on an item in the suggestion bar or a backspace as a keystroke. A high KSPC means that a large number of actions were performed.

KSPC is dependant upon the keyboard's input method. When compared to a baseline in each modality, KSPC is 1 for tap typing and 0.2 for gesture typing (since 1 stroke / 5 chars per

word = 0.2). A lower KSPC indicates the user is taking advantage of completions and predictions while a higher KSPC indicates the user "wasting" keystrokes in correcting errors.

CER (character error rate) compares the resulting output string to the goal string. Unlike KSPC, it does not reflect in-process errors and corrections, but instead only considers the final string committed by the user. CER is calculated by taking the minimum edit (Levenshtein) distance between the goal string and the committed string, divided by the number of characters in the goal phrase.

USER STUDY RESULTS

The user study presented in this paper is not a factorial psychological experiment. We therefore use basic descriptive statistics in our reporting.

Test 1 (gesture typing): Table 1 shows the mean (STD), median, minimum, and maximum WPM, CER, and KSPC across all 18 participants in this test. Gesture speeds averaged 24.0 WPM at 3.7% error rate.

	Mean (STD)	Median	Min	Max
WPM	24.0 (5.8)	24.3	10.3	32.3
CER (in %)	3.7 (3.6)	3.2	0.0	12.4
KSPC	0.4 (0.2)	0.4	0.3	1.0

Table 1. Gesture typing WPM, CER, and KSPC results.

Test 2 (tap typing): Table 2 shows the mean (STD), median, minimum, and maximum WPM, CER, and KSPC across all 18 participants in this test. Tap speeds averaged 22.0 WPM at 1.5% error rate.

	Mean (STD)	Median	Min	Max
WPM	22.0 (2.4)	22.3	16.1	27.3
CER (in %)	1.5 (2.4)	0.6	0.0	10.2
KSPC	1.5 (0.2)	1.5	1.1	1.9

Table 2. Touch typing WPM, CER, and KSPC results.

A one-way between-subjects ANOVA did not show that the overall effect of gesture compared to tap on WPM was significant ($F(1,36) = 1.59, p = 0.21$). It did show that the effect on CER and KSPC was significant ($F(1,36) = 4.49, p = 0.04$ and $F(1,36) = 248.60, p < 0.001$).

DISCUSSION

Although as an initial prototype WatchWriter decoding parameters were not optimized for the watch size, we see very promising results for both gesture and tap typing on a watch. Before we tried it, developing and using a regular single-step Qwerty on-screen keyboard on a watch initially seemed impractical even to us. Other researchers have also thought it impractical, as reckoned in a CHI 2015 paper "It is doubtful that the QWERTY soft keyboard should be adopted on smartwatches" [5]. The basic results from this paper counter that intuition. Not only can such keyboards be made to work, but they worked markedly better than the previous smartwatch keyboard techniques reported in the literature.

Without any prior experience, the average first 10 minutes typing speed was 22 WPM at 1.5% error for tap, and 24 WPM at 3.5% error for gesture. This is markedly higher than Zoomboard, SplitBoard, and ZShift which, as mentioned in Related Work, range from 9.1–16.3 WPM at 0.9%–1.4% error. This is also higher than SwipeBoard's 19.6 WPM after 2 hours of practice at error rates of 4.2% for their first step and 13.3% for their second step.

To investigate this further, we can contrast our results to Reyal et al's similar study of the performance of Google Keyboard on Android phones [11]. In their experiment with conditions most similar to ours, they found a mean WPM of 29.4 for tap typing and 25.8 for gesture typing.

CONCLUSIONS AND FUTURE WORK

WatchWriter demonstrated that the single-step tap and gesture typing can be quite effective even on a small watch form factor, thanks to the combination of auto-correction and prediction capabilities already developed on smartphones and the human motor control flexibility and adaptability.

Whether the level of performance demonstrated in our study is practically acceptable to end users in everyday use remains an open question and calls for further work, including:

Alternative Keyboard Layouts: the small size, increased ambiguity and often rounded shape of watches call for research on keyboard layouts alternative to Qwerty.

OOV Words: We have not explored entering out-of-vocabulary words. Predictive tap typing on Qwerty generates a literal string as well as predictions, and our initial tests show that it is possible to type OOV words without frustration.

Non-alphabetical Characters: This paper also did not focus on entering non-alphabetical characters (such as punctuation, numbers, etc.), and future work could explore the best way to enable them. Non-alphabetical characters are likely typed less often than words, so implementing this feature could be as simple as a drop down menu.

Alternate Watch Shapes And Sizes: Our evaluation used just a single round watch. Future studies could explore predictive tap and gesture typing on a wider variety of shapes and sizes to see how they impact performance.

Deeper Modelling: Due to the scope of this paper, we did not build models that characterize the differences in tap and gesture behavior on watches in comparison to on larger devices, such as smartphones or tablets. Future research in that direction could inform how keyboard decoding and language models could be optimized for a watch.

ACKNOWLEDGEMENTS

We would like to thank Kurt Partridge, Xiaojun Bi, and other colleagues at Google for for their valuable insights and contributions to this work. We would also like thank the CHI 2016 papers committee for helping to narrow the scope and sharpen the contribution of this paper.

REFERENCES

1. Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto International Conference*. IEEE, 100–105.
2. Shiri Azenkot and Shumin Zhai. 2012. Touch Behavior with Different Postures on Soft Smartphone Keyboards. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '12)*. ACM, New York, NY, USA, 251–260. DOI: http://dx.doi.org/10.1145/2371574.2371612
3. Xiaojun Bi, Yang Li, and Shumin Zhai. 2013. FFitts Law: Modeling Finger Touch with Fitts' Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 1363–1372. DOI: http://dx.doi.org/10.1145/2470654.2466180
4. Xiang 'Anthony' Chen, Tovi Grossman, and George Fitzmaurice. 2014. Swipeboard: A Text Entry Technique for Ultra-small Interfaces That Supports Novice to Expert Transitions. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 615–620. DOI: http://dx.doi.org/10.1145/2642918.2647354
5. Jonggi Hong, Seongkook Heo, Poika Isokoski, and Geehyuk Lee. 2015. SplitBoard: A Simple Split Soft Keyboard for Wristwatch-sized Touch Screens. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 1233–1236. DOI: http://dx.doi.org/10.1145/2702123.2702273
6. Andreas Komninos and Mark Dunlop. 2014. Text Input on a Smart Watch. *IEEE Pervasive Computing* 13, 4 (2014), 50–58. DOI: http://dx.doi.org/10.1109/MPRV.2014.77
7. Per-Ola Kristensson and Shumin Zhai. 2004. SHARK2: A Large Vocabulary Shorthand Writing System for Pen-based Computers. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology (UIST '04)*. ACM, New York, NY, USA, 43–52. DOI: http://dx.doi.org/10.1145/1029632.1029640
8. Luis A. Leiva, Alireza Sahami, Alejandro Catala, Niels Henze, and Albrecht Schmidt. 2015. Text Entry on Tiny QWERTY Soft Keyboards. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 669–678. DOI: http://dx.doi.org/10.1145/2702123.2702388
9. I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase Sets for Evaluating Text Entry Techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03)*. ACM, New York, NY, USA, 754–755. DOI: http://dx.doi.org/10.1145/765891.765971
10. Stephen Oney, Chris Harrison, Amy Ogan, and Jason Wiese. 2013. ZoomBoard: A Diminutive Qwerty Soft Keyboard Using Iterative Zooming for Ultra-small Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2799–2802. DOI: http://dx.doi.org/10.1145/2470654.2481387
11. 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. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 679–688. DOI: http://dx.doi.org/10.1145/2702123.2702597
12. 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 SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 113–120. DOI: http://dx.doi.org/10.1145/642611.642632
13. 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. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 659–668. DOI: http://dx.doi.org/10.1145/2702123.2702135