Word Clarity as a Metric in Sampling Keyboard Test Sets

Xin Yi[1], Chun Yu[1†], Weinan Shi[1], Xiaojun Bi[2], Yuanchun Shi[1]
[1]Key Laboratory of Pervasive Computing, Ministry of Education
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
[2]Department of Computer Science, Stony Brook University, Stony Brook, NY, United States
{yix15,swn16}@mails.tsinghua.edu.cn {chunyu,shiyc}@tsinghua.edu.cn xjunbi@gmail.com

ABSTRACT
Test sets play an essential role in evaluating text entry techniques. In this paper, we argue that in addition to the widely adopted metric of bigram representativeness and memorability, *word clarity* should also be considered as a metric when creating test sets from the target dataset. Word clarity quantifies the extent to which a word is likely to confuse with other words on a keyboard. We formally define word clarity, derive equations calculating it, and both theoretically and empirically show that word clarity has a significant effect on text entry performance: it can yield up to 26.4% difference in error rate, and 25% difference in input speed. We later propose a Pareto optimization method for sampling test sets with different sizes, which optimizes the word clarity and bigram representativeness, and memorability of the test set. The obtained test sets are published on the Internet.

Author Keywords
Text entry evaluation; sampling; word clarity; phrase set.

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation: User Interfaces - Input devices and strategies

INTRODUCTION
Due to the "fat-finger problem" [35] and lack of tactile feedback, text entry has long been a challenge for touchscreen devices [2]. So far, a considerable amount of effort has been taken to address this problem. As with any mature research field, establishing a sound evaluation methodology plays a key role in advancing the status quo. In text entry, a well-accepted evaluation methodology is conducting user studies, in which participants are instructed to transcribe a set of phrases while the speed and error rate are measured.

One core component of a user study is the test set. As we will see in this paper, in a typical study, an average of 28 phrases are tested for each condition and participant. Corresponding to the trade-off between internal validity and external validity, there exist two major approaches for selecting these phrases: 1) randomly pick different phrases for different participants and 2) sample some phrases from well-known datasets (e.g. Mackenzie and Soukoreff dataset [25], EnronMobile dataset [39]), and use them for all participants. Focusing on the second approach, researchers have proposed several criteria to ensure that the sampled test sets truly represent the target language. For example, the bigram probability of the test set should be representative of that of the target dataset [31], and its memorability should also be maximized [23].

Besides the two metrics mentioned above, we argue that *word clarity* of the target dataset should also be represented in the test set. Word clarity quantifies the extent to which a word is likely to confuse with other words on a keyboard, or in other words, the difficulty to correctly enter a word on a keyboard. For example, "in" is often mistyped as "on", because the letter 'i' and 'o' are close to each other and both "in" and "on" are very common words. On the contrary, "plus" is easy to enter because it has no strong distractors.

In literature, the concept of word clarity has been noticed by some researchers [10, 21, 37]. However, none of them has explained the probabilistic interpretation of their calculation, nor have they formally investigated the effect of word clarity on text entry measurements. As a result, there is little quantitative result about word clarity beyond the "intuitiveness", and it is difficult for researchers to leverage the knowledge to improve the methodology of text entry evaluation.

In this paper, we defined word clarity and investigated its effect on input speed and error rate. Based on the results, we proposed a Pareto optimization approach to include word clarity in test set sampling. Specifically, we have made the following two contributions: 1) We formally defined word clarity, and derived equations calculating it from probabilistic theory. We theoretically and empirically showed that word clarity of the test words has a significant effect on the measured text entry speed and error rate: it can yield up to 25% difference in input speed, and 26.4% difference in error rate; 2) We proposed a Pareto optimization method for sampling test sets, which optimizes the word clarity and bigram representativeness, as well as memorability of the test sets. We have applied the proposed method to obtain test sets from the Mackenzie and Soukoreff phrase set [25], and published them on the Internet [3] to immediately benefit text entry researchers.

† denotes the corresponding author.

CHI 2017, May 06-11, 2017, Denver, CO, USA

DOI: http://dx.doi.org/10.1145/3025453.3025701

RELATED WORK

Smart Touch Keyboards

The classical statistical decoding algorithm of input correction was proposed by Goodman et al. [16], which calculates the likelihood of a word W within a pre-defined dictionary given an input sequence I as:

$$P(W|I) \propto P(I|W) \times P(W) \quad (1)$$

where $P(I|W)$ describes the noise in users' typing behavior, and $P(W)$ corresponds to the frequency of the word. After that, numerous smart keyboards have been proposed for various scenarios (e.g. tablet [11, 21], phone [14, 41], and smartwatch [40]). Noticeably, QWERTY still stands as one of the most popular layouts for smart keyboards. Therefore, we focus our review on smart keyboards with QWERTY layout.

Based on the classical statistical decoding algorithm, Findlater et al. [11] adapted the underlying key-press classification models to improve the ten-finger typing performance. Vertanen et al. proposed VelociTap [40], a decoder that incorporates a probabilistic keyboard model, a character language model and a word language model. Kristensson et al. [21] proposed a keyboard that uses geometric pattern matching to perform word-level correction. Some smart keyboards have leveraged additional information. Goel et al. [14] used accelerometer data to correct typing errors resulting from walking. Later, Goel et al. proposed ContextType [15], which combines users' posture-specific touch pattern information with a language model to classify users' touch events as pressed keys. Weir et al. [41] combined pressure information and Gaussian Progress regression within a probabilistic decoder.

The algorithm of smart keyboards works by searching for legal words within a predefined dictionary that most like the input. Therefore, input correction is in essence a classification problem. This inspired us to take *word clarity*, which quantifies the "similarity" between the word and other words, into consideration when evaluating smart keyboards.

Word Clarity

When evaluating smart keyboards, word clarity is an important factor that has been noticed by several researchers [10, 21, 37]. Generally, word clarity describes the extent to which a word is likely to confuse with other words. Kristensson et al. [21] noticed that the "difficulty" of the task words affects the performance of the keyboard. However, they did not further investigate this effect. Dunlop et al. [10] and Smith et al. [37] used word clarity in optimizing keyboard layout for touch and gesture typing respectively.

In existing works, researchers tended to calculate word clarity using the location and shape of the word on the keyboard layout. Kristensson et al. [21] did not explicitly quantify word clarity. However, they noticed that some word pairs are difficult to distinguish because they are close neighbors on the keyboard. Dunlop et al. [10] calculated the "tap ambiguity" based on "badgrams" to quantify the clarity of a word. Smith et al. [37] defined gesture clarity as the pairwise Manhattan distance between sampled points on each gesture.

We see two limitations in existing works: 1) the calculation of word clarity was not derived from principled probabilistic theory. Consequently, the calculation of word clarity varied among researchers; 2) there lacks a formal investigation of the effect of word clarity on text entry measurements.

Phrase Sets for Text Entry Experiments

In early days, there was no widely agreed upon standard on the selection of appropriate phrases for transcription tasks. Therefore, researchers have used texts from a wide range of resources (e.g. western novel [18], Linux operating system [17], and news [43]). In order to improve this situation, Mackenzie et al. [25] proposed a standard phrase set that contains 500 phrases. And Vertanen et al. [39] proposed the EnronMobile phrase set, which was optimized in terms of both memorability and bigram probability.

Due to the limited number of task phrases in experiments, researchers usually use a randomly sampled subset of the mainstream phrase sets (e.g. [40, 41]). Some researchers also added pangrams to the task phrases to ensure the coverage of all letters (e.g. [11, 15, 21]). To minimize sampling error, researchers have proposed advanced methods to generate better phrase sets. Paek et al. [31] proposed a procedure that can choose the best phrase set from a number of randomly generated samples based on relative entropy. Leiva et al. [23] emphasized the memorability of phrases, and proposed a method for sampling memorable and representative phrase sets based on a multiple regression model.

Some researchers have compared the performance of different phrase sets and sampling methods. Kristensson et al. [20] compared five publicly-available phrase sets and two task presentation styles. They found that different phrase sets yield statistically significant differences in terms of both entry and error rates. Later, Sanchis et al. [33] compared three automated phrase sampling methods in ten languages, and found that MEMREP [23] outperforms RANDOM and NGRAM [31].

WORD CLARITY OF INDIVIDUAL WORD

Although it is agreed that word clarity describes the extent to which a word is to confuse with others, the calculation of word clarity in existing works varies. In this section, we first introduce the calculation of word clarity and its probabilistic interpretation. We then investigate the effect of word clarity on the measured error rate through simulation.

Calculating Word Clarity

A key concept in calculating word clarity is the "distance" between words. In existing works, it is usually calculated according to the spatial distance between words on the keyboard layout (e.g. [37]). Similarly, we calculated the distance between two words as:

$$dis(A,B) = \frac{1}{S_{key}^2} \times \sum_{i=1}^{n} \|A_i - B_i\|_2^2 \quad (2)$$

In Equation 2, A and B are two words whose length are both n. A_i and B_i are the 2D key centers of the ith character in A and B respectively, and $||\cdot||_2$ denotes the Euclidian norm. S_{key} is the size of each key used for normalization. We assume that

users can input the correct number of points when entering the target word, hence we only consider word pairs with identical lengths. The smaller $dis(A,B)$ is, the more similar A and B are, and it is more likely that the two words may confuse with each other.

Given the distance between word pairs, we then define the clarity of a word W in the same way as existing works [37]:

$$clarity(W) = \min_{X \in L(n)-W} dis(W,X) \quad (3)$$

where n corresponds to the length of W, $L(n)$ denotes the set of all words in the dictionary whose lengths are n. In Equation 3, $clarity(W)$ can be interpreted as the minimum distance between W and all other words in the dictionary with identical lengths. The lower the clarity is, the more likely W may be to confuse with other words. Particularly, we define $clarity(W)$ to be infinity if W is the only word in $L(n)$, because there are no other words that can be confused with W.

Probabilistic Interpretation

Note that Smith et al. [37] also presented a formula for calculating word distance:

$$dis(A,B) = \frac{1}{n} \times \sum_{i=1}^{n} \|A_i - B_i\|_2 \quad (4)$$

However, there are two major differences between Equation 2 and Equation 4: 1) Instead of using the Euclidian norm, we use the squared Euclidian norm. Accordingly, we use S_{key}^2 to normalize the result with regard to keyboard size; 2) We do not normalize the result with regard to the length of the word (n). We now prove this modification is helpful to make the calculation result more interpretable. We denote λ as the probability that a user intends to enter word A, but generates the input that corresponds to word B. We can calculate λ as:

$$\lambda = \prod_{i=1}^{n} P(B_i|A_i) \quad (5)$$

where n is the length of both A and B. It is widely accepted that the touch endpoint on touchscreen keyboards follows a 2D Gaussian distribution (e.g. [4, 13, 16]). For simplicity, we assume the standard deviation in x and y dimensions are identical, therefore

$$P(B_i|A_i) = \frac{1}{2\pi\sigma^2} \exp\{-\frac{1}{2\sigma^2}[(B_{ix}-A_{ix})^2 + (B_{iy}-A_{iy})^2]\} \quad (6)$$

Combining Equation 5 and Equation 6, we have

$$\lambda = (\frac{1}{2\pi\sigma^2})^n \exp(-\frac{1}{2\sigma^2}\|A_i - B_i\|_2^2) \quad (7)$$

Comparing Equation 2 and Equation 7, we can see that by using the squared Euclidian norm, our definition of "distance" is indicative of the probability that users generate ambiguous input. Essentially, this square form reflects the 2D Gaussian noise in users' input. According to symmetry, this is also true when A and B are exchanged with each other. Finally, we normalize the distance metric according to keyboard size by dividing the S_{key}^2 term, which makes the result consistent across different sizes of keyboards.

Keyboard Layout and Language Model

Equation 2 and Equation 3 suggest that the value of word clarity is specific to a particular keyboard layout and a dictionary. In order to maximize the external validity of our result, in this paper, we calculated word clarity using the standard QWERTY keyboard layout, which is common across most Android keyboards (e.g. Nexus 5) and in existing works [13, 37]. The width and the height of individual keys is 6.16mm and 9.42mm respectively, with no margin between keys.

Meanwhile, we used the *Enron Corpus* [19] as our dictionary. The Enron Corpus is a large set of emails that were generated by the employees of the Enron Corporation, which consists of over 600,000 messages from 1999 to 2002. In the field of text entry, the Enron Corpus has been adopted as language model by many researchers [13, 31, 39]. We chose it for two reasons: 1) it consists of text that are generated in real world human communication without privacy problem; 2) it consists of a relatively large body of text, which is essential to fit the huge body of the English language.

Unfortunately, the Enron Corpus was inadequate in its raw form. Therefore, we performed preprocessing to get clean text that is generated by human (rather than generated by the machine), and to filter out words that are illegal. Our progressively preprocessing of the Enron Corpus contains deduplication, removing attached text, email address, URLs and all non-alphabetical characters, etc.

As addressed by Fowler et al. [13], extracting clean, human-generated text from the Enron Corpus is not easy. We therefore verified our results with Mackenzie et al.'s work [25]. In the final corpus, there are totally 53,226,381 words, 253,165,481 characters, with 148,565 distinct words. The mean length of word is 4.76. Table 1 shows the most frequent letters and words in our processed corpus. Not surprisingly, "e" is the most frequent letter, and "the" is the most frequent word. The correlation with English using *AnalysePhrase.java* [25] was 0.942. According to these results, we believe our processed corpus is representative of the English language.

Letter	Frequency	Probability
e	30,958,000	0.1223
t	22,245,950	0.0879
a	20,856,668	0.0824
o	19,735,975	0.0780
n	18,723,388	0.0740

(a)

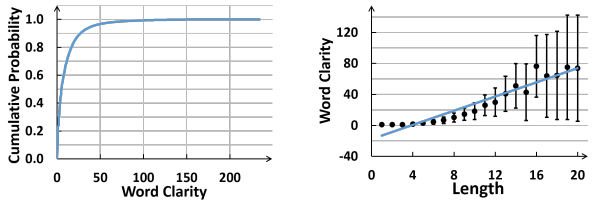
Word	Frequency	Probability
the	2,485,564	0.0467
to	1,600,379	0.0301
and	1,131,430	0.0212
of	1,046,738	0.0197
a	888,908	0.0167

(b)

Table 1: Most frequent letters and words with their frequencies in our processed corpus.

Distribution of Word Clarity

Figure 1a shows the *Cumulative Distribution Function (CDF)* of the clarity of all words in the corpus. Of all 148,565 distinct words, most words yield a relatively small clarity. The mean clarity is 11.2 (SD = 17.0), and 77.5% of the words have a clarity below 15.0. On the other hand, there are also some words with very high clarity, yielding an overall range from 1.0 to 234.5.



(a) (b)

Figure 1: (a) CDF of the clarity of all words in the corpus; (b) Linear fitting of word clarity to word length, black bar shows one standard deviation.

Figure 1b shows the clarity of words with different lengths. Generally, longer words tend to have higher clarity. A linear fitting between word length and average clarity yield an R^2 of 0.91. On the other hand, the R^2 value fitted on all words dropped to 0.49, suggesting that word length *per se* is not sufficient to model word clarity.

EFFECT OF WORD CLARITY ON ERROR RATE

The task of smart keyboards is to find the word in a dictionary that is "most like" the input. Therefore, even under the same level of input noise, words with higher clarity may still yield higher accuracy than those with lower clarity. However, to our knowledge, the effect of word clarity on accuracy has never been verified. In this section, we verify through simulation whether the clarity of the target word indeed affects the accuracy of smart keyboards. Additionally, as the keyboard size varies significantly across platforms (e.g. smartphone and smartwatch), we also investigate the effect of keyboard size.

Simulation Design

The basic workflow of our simulation procedure is to generate simulated keyboard taps for words with different clarity, use a smart keyboard algorithm to recognize the target word according to the noisy input, then evaluate the output.

We used the same keyboard layout as existing works [13, 37]. Three levels of keyboard sizes were tested, with the key widths being 6.16mm, 4.8mm and 2.4mm respectively. These sizes approximate the size of keyboards on smartphones [4, 6] and smartwatches [22]. For each keyboard size, we tested all distinct words in the Mackenzie and Soukoreff phrase set [25], as it is one of the most widely-adopted phrase sets in current studies. And for each word, we ran 1,000 independent iterations of simulation. Totally, we tested 3 *sizes* $\times$ 1163 *words* $\times$ 1000 *iterations* $=$ 3489000 *iterations*.

We modelled the distribution of touch location using 2D Gaussian distribution [13]. The mean of the distribution was set at the center of each key, which is a model simplification for general typing tasks without biasing towards a particular touch finger or angle. Previous work [4] also found that the magnitude of the offset in general is small ($\sim$0.5mm) and the average offset across all the conditions are close to the target center. The x and y standard deviation in the big size was set to be 1.97mm and 1.88mm respectively based on Azenkot et al.'s results [4]. For the middle and small sizes, the standard deviations were set to be 1.88 and 1.68mm in both dimensions according to Bi et al.'s findings [6]. These values yielded a per-tap error rate of 61.9%, 24.2% and 12.9% for increasing sizes respectively. Note that we have normalized $dis(w_1, w_2)$ according to the key size, therefore this would not affect the result of word clarity. We used the classical Bayesian model as the keyboard algorithm (see Equation 1). $P(I|W)$ was calculated using the Gaussian distribution described above, and $P(W)$ was from our language model.

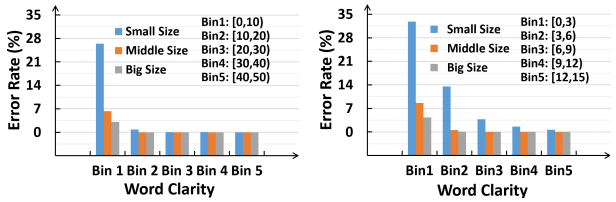
Results

Considering that character-level metrics are not suitable for smart keyboards, which work at the word level rather than literally output every letter typed, some researchers have proposed word-level metrics that are helpful in smart keyboard evaluation (e.g. *word score* [5]). Here, for simplicity, we refer to the word-level error rate as:

$$Error\ Rate = (1 - \frac{N_{correct}}{N_{total}}) \times 100\% \quad (8)$$

where $N_{correct}$ and N_{total} denotes the number of correct words and the total number of the transcribed words respectively.

There are totally 1,163 distinct words in the Mackenzie and Soukoreff phrase set [25], with clarity ranging from 1.0 to 49.9. We evenly split the range of word clarity into 5 bins, and calculated the average word-level error rate of the words in each bin (Figure 2a). Noticeably, 91.3% of the words has a clarity less than 15.0. Therefore, we also performed the same analysis for these words separately (Figure 2b). We did not control the number of words in each bin to be equal, as there are a large amount of words that has identical word clarity. For example, in Figure 2b, 1584/3186 words has a clarity of 1, which makes them inseparable.



(a) (b)

Figure 2: Word-level error rate for each clarity bin and keyboard size. (a) all 1,163 words; (b) words with clarity less than 15.0.

Figure 2 shows the word-level error rate for each clarity bin and keyboard size. For all sizes, the word-level error rate drops monotonously with increasing word clarity, implying that even under the same level of input noise, words with higher clarity will indeed yield higher accuracy. Besides, the effect of clarity on error rate becomes more pronounced when keyboard size becomes smaller (3.1% to 0.0% for the big size, while 26.4% to 0.0% for the small size). Note that the level of input noise also becomes greater compared with the keyboard size for smaller keyboards, this results thus highlights the need for considering word clarity for imprecise input scenarios.

Meanwhile, the effect of word clarity seems more significant when clarity is relatively low. In Figure 2a, even for small size, the average error rates of bin2 to bin5 are close. However, in Figure 2b, there is an obvious trend that error rate dropps monotonously with increasing clarity. It is worth mentioning that most of the words (77.5% in the whole corpus and 91.3% in the Mackenzie and Soukoreff phrase set [25]) have a relatively low clarity (< 15.0), therefore we consider this "small-scale effect" important.

TYPING PERFORMANCE ON DIFFERENT PHRASE SETS

The results in the previous section highlight the need for considering the clarity of task words when measuring the word-level accuracy of smart keyboards, especially when the level of input noise is relatively large compared with the keyboard size (e.g. on smartwatches). In this section, we carried out a user study where participants completed real text entry tasks. Our goal is to verify whether testing phrase sets with different word clarity would really bias the measured speed and accuracy. Similar as the previous study, we tested two different sizes of keyboards: phone-sized and watch-sized.

Participants

We recruited 12 participants from the campus (8 male, 4 female), with an average age of 21.8 (SD = 2.3). All participants regularly used a QWERTY keyboard on their smartphones. Each participant was compensated $20.

Apparatus

As we were interested in users' typing performance on different sizes of keyboards, we used two kinds of apparatus: A Nexus 6P phone for the phone-sized keyboard, and a MOTO 360 smartwatch for watch-sized keyboard. The Nexus 6P phone has a 5.7 inch screen, with a ppi of 515. The MOTO 360 smartwatch has a 1.56 inch round screen, with a ppi of 205. Both apparatus report the location in pixel level and timestamp when a touch event occurs (e.g. down, move, up).

Experiment Design

We used a two-factor within-subjects design. We tested two keyboard sizes: phone-sized and watch-sized. For the phone-sized keyboard, we used the *big size* layout in the previous section. For the watch-sized keyboard, in pilot study, uses commented that 2.4mm keys was too small for typing. Therefore, we set the keys to be 3mm×3mm, with no margin between keys. The keyboard algorithm was the same as the *big size* and *small size* keyboard in the previous section respectively. The only difference is that for real time performance, we only used the top 15,000 words in the dictionary as the language model. As noted by Nation et al. [29], this was sufficient to cover about 97.8% of daily English words.

We manually chose three phrase sets from a number of subsets randomly sampled from the Mackenzie and Soukoreff phrase set [25], denoted as *Easy*, *Medium* and *Hard*. Each phrase set has 20 distinct phrases. The average word clarity of all words in *Easy*, *Medium* and *Hard* was 9.96, 6.02 and 1.08 respectively. We expected that *Easy* would yield the highest text entry speed with lowest error rate, while *Hard* would led to the lowest text entry speed with highest error rate.

Procedure

Participants completed two sessions of text entry tasks, each corresponding to a keyboard size. In each session, they completed three blocks of text entry tasks, corresponding to *Easy*, *Medium* and *Hard* respectively. The order of sessions and blocks were counterbalanced. Participants were seated during the experiment. After a three-minute warm-up, they were asked to type "as quickly and as accurately as possible". As our goal is to investigate the relative performance across different conditions, we asked all participants to type with their index finger to avoid bias that arise from different typing postures. During typing, the keyboard shows three candidate words (see Figure 3). Upon selection, the keyboard automatically appends a space to the word. Users can also press backspace or swipe left to correct the entered text. A two-minute break was enforced between each block.

(a) Smartwatch interface (b) Phone interface

Figure 3: Experiment platform.

Results

Text Entry Speed

We calculated the text entry speed following Mackenzie [1]:

$$WPM = \frac{|T|-1}{S} \times 60 \times \frac{1}{5} \quad (9)$$

where T is the target string and S is the elapsed time in seconds from the first to the last touch in the sentence. Figure 4 shows the average text entry speed for each condition.

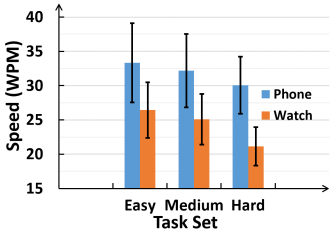


Figure 4: Text entry speed for each keyboard size and phrase set, black bar shows one standard deviation.

For phone-sized keyboard, the speed for *Easy*, *Medium* and *Hard* was 33.3 WPM (SD = 5.8), 32.2 WPM (SD = 5.4) and

30.0 WPM (SD = 4.2) respectively. Generally, this speed is consistent with the index-finger text entry speed on smart phones [4]. Interestingly, average text entry speed increased monotonously with word clarity, with the speed of *Easy* being 11% faster than that of *Hard*. RM-ANOVA found a significant effect of clarity ($F_{2,22} = 13.1, p < .001$), confirming that task clarity has effect on the measured text entry speed.

As expected, the speed on the watch-sized keyboard was slower. The speed for *Easy*, *Medium* and *Hard* was 26.4 WPM (SD = 4.0), 25.1 WPM (SD = 3.7) and 21.1 WPM (SD = 2.8) respectively. RM-ANOVA also found a significant effect of clarity ($F_{2,22} = 57.1, p < .0001$). Interestingly, the effect of clarity was much stronger than phone-sized keyboard, with the speed for *Easy* being 25% faster than that for *Hard*. This phenomenon is consistent with our simulation result in the previous section, which may be because of the relatively larger level of input noise.

Error Rate

We measured error rate using CER [38], which is the minimum string distance between the target string and the final transcribed string, divided by the length of the target string. Figure 5 shows the average error rate in each condition. Consistent with existing works, subjects tended to fix most of the errors and left few in the final submitted string [24, 38, 42]. The error rate of all six conditions were below 0.9%.

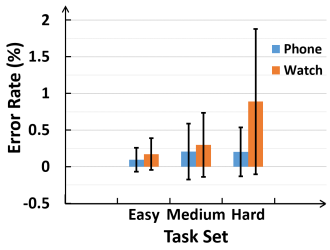


Figure 5: Error rate for each keyboard size and phrase set, black bar shows one standard deviation.

For both phone-sized and watch-sized keyboard, the average error rate increased monotonously as word clarity drops. And again, the effect of word clarity was stronger for watch-sized keyboard than phone-sized keyboard. The average error rate for *Hard* was 0.5 times higher than that for *Easy* in the phone-sized keyboard, but 4.2 times higher for watch-sized keyboard. RM-ANOVA found no significant effect of clarity ($F_{2,22} = 0.60, p = .56$) for phone-sized keyboard, probably due to the overall high prediction accuracy. While for the watch-sized keyboard, a significant effect of clarity was found ($F_{2,22} = 5.27, p < .05$).

PHRASE SET SAMPLING PROBLEM

So far, we have showed that the clarity of the chosen task phrases would have effect on text entry studies in terms of both the measured speed and accuracy. Therefore, it is necessary to take word clarity into consideration when sampling phrase sets. In this section, we formalize the phrase sampling problem, and introduce the metrics that we consider in sampling proper phrase sets for text entry experiments.

Phrase Set Sampling Problem

When evaluating the performance of text entry techniques, a widely adopted way is to recruit participants in a text entry experiment, in which they are asked to transcribe some task phrases while input speed and accuracy are measurements. The advantage of the transcription task is that it strengthens the internal validity. First, as all participants write the same text, this removes the variance that might occur due to participants writing widely varying texts. Second, a transcription task does not require participants to think of something to write, which reduces the variance in text entry speed. Third, this allows results to be reproducible, and facilitates the comparison of different text entry techniques.

However, the downside of a transcription task is its low external validity. That is, due to the limited number of task phrases used in the experiment, the measured performance of the text entry methods is hard to be generalized as the performance in realistic settings. The tradeoff between internal validity and external validity in text entry experiments has been extensively discussed by many researchers [20, 23, 25, 33, 39].

In this paper, we focus on optimizing the external validity of a transcription task by choosing the "right" phrases. We define the problem of phrase sampling as: Given S as a set of phrases that are good candidates for conducting text entry experiments (in this paper, the Mackenzie and Soukoreff phrase set [25]), find the optimal subset of S (denoted as $\hat{S}$) that is most representative of the target language (e.g. English). In this paper, we used our processed Enron Corpus [19] as described previously to approximate the characteristics of the English language (denoted as U).

Optimization Metrics

When sampling phrase sets for text entry experiments, *memorability* and *representativeness* are two metrics that have been widely adopted by researchers [23, 25, 33, 39]. In transcription tasks, it is important that phrases should be memorable in order to minimize the variance of additional cognitive processing time. Meanwhile, as speed and accuracy are the most important measurements in text entry experiments, we hope that the measured speed and accuracy of a smart keyboard on $\hat{S}$ should be representative of that on U.

In the previous section, we have proved that the clarity of the task phrases affects the measured speed and accuracy. Meanwhile, *bigram probability* has been extensively used for both predicting the text rate [8, 10, 26, 36] and for designing representative phrase sets [31, 39]. Researchers usually calculate the movement efficiency ($\overline{MT}$) as the sum of Fitts' law [12] movement time (T_{ij}) between all pairs of letters (bigrams) weighted by bigram probabilities P_{ij} calculated from a language corpus:

$$\overline{MT} = \sum_i \sum_j P_{ij} \times T_{ij} \tag{10}$$

where

$$T_{ij} = a + b log_2(\frac{A_{ij}}{S_{key}} + 1) \tag{11}$$

A_{ij} is the distance between key i and key j, S_{key} is the size of the key, a and b are coefficients. Based on these facts, we

ground the notion of representativeness in terms of word clarity and bigram probability. Besides, we take the memorability of the sampled phrases as the third metric. We now discuss the calculation of the three metrics in detail.

Word Clarity Metric

We measure the similarity between the distribution of word clarity of $\hat{S}$ and that of U based on the *Kolmogorov-Smirnov test (K-S test)* [28]. The K-S test is a widely adopted non-parametric test for comparing the distribution of two samples in statistics. It makes no assumptions about the probability distributions of the variables being assessed, making it suitable for a wide range of applications. We defined $D(\hat{S},U)$ as a new metric quantifying the difference between the word clarity distribution of $\hat{S}$ and U, which could be calculated as:

$$D(\hat{S},U) = \sup_t |F_{\hat{S}}(t) - F_U(t)| \tag{12}$$

where

$$F_U(t) = \frac{1}{n}\sum_{i=1}^{n} 1\{x_i \le t\} \qquad x_i \in U \tag{13}$$

In Equation 13, n is the size of U, $F_U(t)$ is a step function that describes the cumulative frequency of x. That is, for any specific value of t, the value of $F_U(t)$ indicates the proportion of individuals in U having measurements less than or equal to t. In Equation 12, $F_{\hat{S}}(t) - F_U(t)$ quantifies the aggregated word clarity of $\hat{S}$ with respect to that of U. The smaller this value is, the easier $\hat{S}$ is compared with U. For example, in the previous experiment, the value of *Easy*, *Medium* and *Hard* was -0.29, -0.17 and 0.29 respectively. Accordingly, the $D(\hat{S},U)$ of the three phrase sets was 0.29, 0.17 and 0.29 respectively.

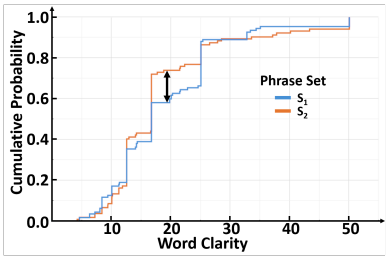


Figure 6: Illustration of the word clarity metric. Blue and orange lines correspond to the empirical distribution function of phrase set S_1 and S_2 respectively, the length of the black arrow corresponds to $D(S_1,S_2)$.

Figure 6 illustrates the calculation of $D(S_1,S_2)$, which can be interpreted as the maximum vertical distance between two empirical distribution functions. It is easy to see that $D(S_1,S_2) \in [0,1]$. $D(S_1,S_2) = 0$ only when the two distributions are identical, and $D(S_1,S_2) = 1$ only when the range of the two distributions do not collapse. The more representative S is with respect to U, the closer $D(S,U)$ will be to zero.

Bigram Probability Metric

The bigram probability table is a 26×26 table describing the probability of each bigram calculated on a corpus. In order to quantitatively measure the similarity between the bigram probability table of $\hat{S}$ and U, we adopted the method of Paek et al. [31], and calculate the representativeness as:

$$D(\hat{S}||U) = \sum_{x,y\in\chi} p_{\hat{S}}(x,y) log_2 \frac{p_{\hat{S}}(x,y)}{p_U(x,y)} \tag{14}$$

where χ is the set of 26 English characters. $p_{\hat{S}}(x,y)$ and $p_U(x,y)$ is the probability of bigram xy in $\hat{S}$ and U respectively. $D(\hat{S}||U)$ can be interpreted as the *relative entropy*, or the *Kullback-Leibler divergence* between the two probability distributions. $D(\hat{S}||U)$ is always non-negative and becomes zero only when the two bigram probability tables are identical. The more representative $\hat{S}$ is with respect to U, the closer the relative entropy will be to zero.

Memorability Metric

Many researchers have highlighted the necessity of considering memorability when designing phrase sets [23, 25, 33, 39]. Leiva et al. [23] found that the memorability of a single phrase $\hat{S}_i$ can be calculated as:

$$CER(\hat{S}_i) = -11.65 + 0.83 \cdot Nw + 0.48 \cdot SDchr + 6.94 \cdot OOV - 1.00 \cdot LProb \tag{15}$$

where Nw is the number of words in the phrase, $SDchr$ is the standard deviation of the number of characters per word. OOV is the ratio of infrequent words. $LProb$ is the logarithm of the probability of the phrase calculated on U. We then calculate the overall memorability of $\hat{S}$ as the average memorability of all phrases in $\hat{S}$:

$$Mem(\hat{S}) = \frac{1}{N}\sum_{i=1}^{N} CER(\hat{S}_i) \tag{16}$$

where N is the number of phrases in $\hat{S}$, and $\hat{S}_i$ is the ith phrase in $\hat{S}$. The smaller $Mem(\hat{S})$ is, the easier it is to remember the phrases in $\hat{S}$.

Metric Normalization

To weigh the three metrics appropriately, we performed an optimization to estimate the minimum and maximum possible values for each metric. We then normalize each of the metric's score in a linear fashion so that the worst possible score is mapped to 0.0 and the best possible score is mapped to 1.0. As Peak et al. [31] demonstrated that the phrase set size affects the characteristics of the phrase set, we chose 4 levels of phrase set size: 20, 40, 80, and 160, and established a normalization system for different phrase sizes respectively.

We used an optimization method combining simulated annealing and local neighborhood search. We performed 20 rounds of optimization for $D(\hat{S},U)$ and $D(\hat{S}||U)$ respectively. Each round starts with a random phrase set, we then ran 2,000 temperatures with 500 iterations in each temperature. For $Mem(\hat{S})$, it is easy to theoretically calculate the maxima and minima by choosing the most/least memorable phrases. The results are shown in Table 2.

The Pareto-Optimization Procedure

We solve the multi-objective optimization problem by performing a *Pareto optimization*, which has recently been used

Phrase Set Size	Clarity Metric		Bigram Metric		Memorability	
	min	*max*	*min*	*max*	*min*	*max*
20	0.011	0.435	0.259	1.820	-0.853	4.513
40	0.007	0.392	0.130	1.295	-0.482	4.204
80	0.004	0.321	0.072	0.879	-0.056	3.780
160	0.002	0.260	0.049	0.552	0.427	3.274

Table 2: Range of metrics used for normalization.

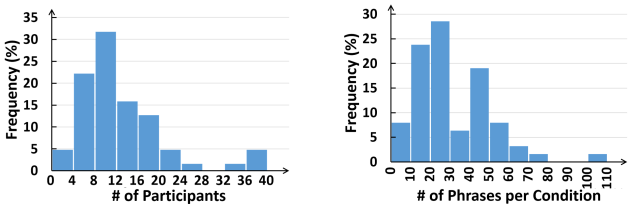
to optimize both keyboard layouts [10, 37] and keyboard algorithms [7]. In this approach, we calculate an optimal set of phrase sets called a *Pareto front*. Each phrase set on the front is called *Pareto optimal*, which means that none of its metrics can be improved without hurting the other scores. Solutions that are not Pareto optimal is called *dominated*, which means that there exists a Pareto optimal solution which is better than it in at least one of the criteria and no worse in the others. Analyzing the Pareto optimal solutions can reveal the tradeoff between multiple objectives. Additionally, the Pareto set provides a broad range of optimal solutions, allowing researchers to choose the one that best matches their preferences.

Our Pareto optimization procedure was similar to that in existing works [7, 37], which is consisted of three phrases: 1) metric normalization; 2) Pareto front initialization; 3) Pareto front expansion. In the first phrase, we used the normalization system in Table 2 as the minimum and maximum possible values for each metric. In the second phase, we evenly chose 49 different weightings for the linear combination of the three metrics, with 4 *size* $\times$ 20 *round* $\times$ 3000 *temperature* $\times$ 1000 *iteration* in each weighting. In the third phase, we performed 500 rounds of expansion to fill out the Pareto front.

SAMPLING PHRASE SETS FOR TEXT ENTRY STUDIES

Task Phrase Sets in Literature

To have an overview of the phrase sets used in existing works, we analyzed all published papers in CHI and UIST from 2003 to 2016 that used the Mackenzie and Soukoreff phrase set [25]. There are totally 44 papers with 63 distinct user studies. Consistently, all the papers used a randomly sampling strategy to get phrase sets for testing. We were interested in the average number of participants in each study, and the number of phrases for each participant and each *condition*, which is the smallest unit in the experiment (e.g. block or session).



(a) (b)

Figure 7: Summarized statistics from published papers. (a) Number of participants; (b) Number of phrases per condition and participant.

Figure 7a shows the distribution of number of participants, and Figure 7b shows the number of phrases per condition and participant. Both metrics roughly follow a Gaussian distribution. The average number of participants in a study is 14.1 (SD = 8.5, median = 12). And the average number of phrases for each condition and participant is 28.3 (SD = 18.0, median = 24). Besides, these two metrics seems independent of each other, with linear regression yielding an R^2 of 0.01. Therefore, for pooled-data analysis across all participants, about $14 \times 28 = 392$ phrases are tested for each condition. And for user-specific analysis, about 28 phrases are tested for each condition and participant.

Validity of Random Sampling

To examine the validity of the widely adopted random sampling approach, we randomly sampled a large number of phrase sets from the Mackenzie and Soukoreff phrase set [25], and check the metrics of the sampled phrase sets. For each level of phrase set size, we randomly sampled 1×10^7 phrase sets. Table 3 shows the simulation result.

Phrase Set Size	Clarity Metric		Bigram Metric		Memorability	
	mean	*std dev*	*mean*	*std dev*	*mean*	*std dev*
20	0.843	0.066	0.736	0.041	0.498	0.052
40	0.850	0.057	0.755	0.032	0.505	0.041
80	0.833	0.051	0.762	0.026	0.506	0.034
160	0.807	0.042	0.741	0.024	0.504	0.029

Table 3: Metrics of the randomly sampled phrase sets.

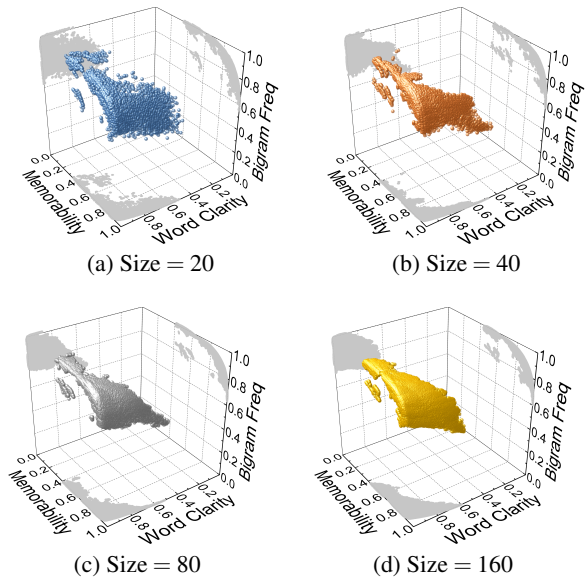
As expected, the standard deviation of all three metrics drop with the increase of phrase set size, suggesting that when more phrases are included, the stability of random sampling can be increased. For all four sizes, the mean clarity metric are between 0.80 and 0.85. However, Table 2 suggested that sometimes the disparity could be very significant (-0.435 to 0.435). Referring to Figure 4 and Figure 5, this corresponds to more than 11% and 25% difference in the measured speed, or 0.5 and 4.2 times difference in the measured error rate, for phone-size and watch-sized keyboard respectively.

Comparatively, random sampled phrase sets yield worse performance in terms of bigram frequency and memorability. The average score is about 0.75 for bigram frequency, and about 0.50 for memorability, which is consistent across different sizes. This suggested that random sampling may not be a good choice to get appropriate test phrase sets. And therefore, a more principled method is needed to get optimal test sets for text entry experiments.

Pareto-Optimized Phrase Sets

Figure 8 shows the outcome of the Pareto optimization, which consists of the final Pareto fronts of optimized phrase sets for different sizes. The Pareto front for increasing sizes is consisted of 3,420, 6,973, 21,247 and 57,292 phrase sets respectively, which are chosen from over 1.1×10^{10} candidate phrase sets. Each Pareto front can be seen as a three-dimensional design space of performance goals that one can choose from for different usage scenarios. Each phrase set on the front is optimal in some tradeoff of the three metrics, and each single set on the

front is better than the others in some way. Generally, a phrase set with higher clarity scores, bigram scores and memorability scores are more apt to exhibit higher external validity in terms of speed and accuracy, and are easier to memorize.



(a) Size = 20 (b) Size = 40

(c) Size = 80 (d) Size = 160

Figure 8: 3D Pareto front for different sizes. Gray dots shows the projection of the points to different planes.

We now highlight phrase sets that serve roughly equal combinations of all the three metrics (those on the 3D Pareto front that are closest to the 45° line through the space). As noted by Dunlop et al. [10] and Smith et al. [37], choosing the solutions that serve each goal on a roughly equal basis is a reasonable approach to get the solutions that can best accommodate a large variety of preferences. We denote each phrase set as "T-<size>", "T" stands for "triple-optimized", "size" takes the value of 20, 40, 80 or 160. Table 4 shows the metric scores of the triple-optimized phrase sets, as well as some optimized phrase sets in previous works. We have published the phrase sets as well as other single-optimized and double-optimized phrase sets online for researchers and designers [3].

Phrase Set	Size	Word / Phrase	Character / Word	Clarity Score	Bigram Score	Memorability Score
T-20	20	4.4	4.6	0.888	0.882	0.887
T-40	40	4.5	4.5	0.901	0.900	0.901
T-80	80	4.6	4.5	0.902	0.903	0.904
T-160	160	4.9	4.6	0.891	0.891	0.891
Enron-Mem1	40	5.4	3.9	0.767	0.697	0.710
Enron-Bi40	40	6.2	4.1	0.720	0.824	0.609
Enron-Mem_Bi	40	6.1	3.9	0.705	0.756	0.609

Table 4: Phrase set metric scores comparison. Shaded rows signify previous phrase sets.

Tradeoff Between Metrics

In Figure 8, the projected points of the Pareto front on three planes are very close to (1.0, 1.0), indicating that these metrics are nearly orthogonal to each other. In fact, in Table 4, the metric scores of the four triple-optimized phrase sets are all close to 0.90. It is a very encouraging finding, which suggests that these three metrics can be optimized simultaneously. This finding remains unchanged as the phrase size varies.

Effect of Phrase Set Size

Interestingly, small-sized triple-optimized phrase sets appears to be subsets of large-sized triple-optimized phrase sets. Table 5 shows the number of identical phrases between each pair of the phrase sets. Approximately, we have T-20 $\subset$ T-40 $\subset$ T-80 $\subset$ T-160. For example, all 40 phrases in T-40 are also in T-80, and 77 out of 80 phrases in T-80 are also in T-160. Therefore, based on our results, one can easily generate triple-optimized phrase sets with any specified sizes. For example, in order to get T-50, one can select 50 phrases from T-80 rather than from all 500 phrases in the raw set.

T-20	/			
T-40	17	/		
T-80	19	40	/	
T-160	20	40	77	/
	T-20	**T-40**	**T-80**	**T-160**

Table 5: Number of identical phrases between each pair of the phrase sets.

Comparison with Other Phrase Sets

To better understand what is possible in the optimization space of phrase set, we compared our results with the Enron-Mobile phrase sets [39]. Table 4 shows the performance of mem1, bi40 and mem_bi phrase sets. These three phrase sets are all consisted of 40 phrases selected from the Enron corpus, and are optimized for memorability, bigram probability or both metrics respectively. As Table 4 shows, our T-40 phrase set significantly outperforms all these three phrase sets on all the three metrics. We believe the key reason is that we considered more factors during optimization, and our optimization procedure is more exhaustive as well as systematic.

Meanwhile, our optimized phrase sets showed significant advantage over random sampling. Compared with Table 3, the triple-optimized phrase sets yielded about 6%, 20% and 80% higher scores in terms of clarity, bigram frequency and memorability respectively. According to Figure 4, Figure 5 and previous results [23, 25, 33, 39]), this could yield effect on the measured speed and error rate in text entry experiments.

DISCUSSION

In this paper, we proposed word clarity as a new metric in sampling keyboard test sets. Although it has been noticed and defined by several researchers (e.g. [37]), we are the first to derive a calculation that has direct probabilistic interpretation. And we both theoretically and empirically verified that, test phrases with different word clarity could yield a difference of 26% in error rate, and 25% in text entry speed.

It is worth noticing that the effect of word clarity was more pronounced on watch-sized keyboard than phone-sized keyboard. We attributed this to the language model in prediction algorithms. For example, although "out" and "our" were close neighbors on the keyboard, they were not the same part of speech. Therefore, they would be easy to distinguish if a powerful language model was employed. In comparison, "in" and

"on" are still hard to distinguish in this case. On phone-sized keyboards, the level of input noise was relatively low. Therefore, language model could compensate most of the input errors, which covered the effect of word clarity. However, on watch-sized keyboard, the level of input noise was much higher. Therefore, the effect of word clarity was still significant. This result suggested that on the one hand, existing results on smart phones are still valid, on the other hand, considering word clarity is important for new scenarios where input precision is low (e.g. smartwatch, distant pointing).

In text entry experiments, there are cases where researchers use the same test phrases for all participants or factor levels (e.g. testing the upper bound of input speed). Comparing with using different phrases for different participants, this approach increased the internal validity of the results. However, there is no principle on how to select these test phrases. In result, the measured results may not be generalized to daily use. The result of this paper can improve the external validity of this case by providing phrase sets that can better fit the target language. And comparing with random sampling, our simulation results also showed that we can achieve higher metric scores regarding external validity.

We employed our approach on the well-known Mackenzie and Soukoreff phrase set [25], and got a set of phrase sets that outperformed existing data sets and the random sampling procedure (see Table 3 and Table 4). Systematically verifying the validity of the method would require users to enter the proposed phrase sets and many phrase sets from existing works (e.g. [39]) and random sampling. However, this would lead to huge amount of experimental work with many controlled factors (word clarity, bigram frequency and memorability), which is impractical. As bigram frequency and memorability have been studied in existing works [23, 31], we focus on phrase sets with different word clarity (see Figure 4 and Figure 5). It is worth noting that our tested phrase sets in this experiment were chosen from several randomly generated phrase sets. Therefore, in real text entry experiments, there could be an even greater difference due to sampling error (see Table 2).

LIMITATIONS AND FUTURE WORK

There are several limitations of this work, which we see as opportunities of future work. First, our calculation of word clarity was restricted to words with identical lengths. There are two reasons for this: 1) Literature [9, 30] shows that the frequencies of insertion/omission errors are only 1% when entering text on phones. Therefore it is safe to assume that most of the time users will input the correct number of touch points; 2) Word clarity is designed to quantify the spatial ambiguity of a test set, independent from keyboard decoders. However, calculating a similar metric for words with unequal lengths requires knowing the penalties that inserting/omitting letters would impose. These penalties are usually tuned for the specific decoder, and vary from keyboard to keyboard. Considering them turns word clarity decoder-dependent.

Second, according to our results, the value and effect of word clarity is affected by interaction modality and the keyboard layout. In this paper, we focus on smart touch keyboards with QWERTY layout, which is the most widely used technique currently. However, it is also worthwhile to validate the results in more scenarios (e.g. smart gesture keyboard [27, 32] and pointing on large wall display [34]). For example, when comparing the similarity of two gestures, Smith et al. [37] proposed the gesture typing word clarity. And one needs to rerun the optimization procedure if the keyboard layout changes. Note that our definition of word clarity is independent of language model, which brings the advantage that our results can also be applied to other corpora different from English.

Third, the goal of our phrase set optimization is to optimize the external validity of the measured results, which assumes that all participants use the same phrase set in the experiment. This could remove the variance due to random sampling. However, in real text entry experiments, this could also bring up carry-over effects (e.g. learning). To minimize the side effects, one can combine the advantage of randomization and optimization by sampling phrase sets from our proposed sets. For example, split T-160 to four sets with 40 phrases each, and assign them to different participants.

CONCLUSION

In this paper, we push the problem of phrase sampling a step further by introducing word clarity, which quantifies how likely a word is to confuse with others. We first derived the calculation of word clarity from probabilistic theory. We then investigated the effect of word clarity on measured speed and accuracy through simulation and user study. Results showed that when the level of input noise is relatively high, word clarity could yield a 25% and 26% difference in measured speed and error rate respectively. Based on these results, we proposed a Pareto optimization procedure to sample phrase sets from the Mackenzie and Soukoreff phrase set, which were optimized in terms of word clarity, bigram frequency and memorability. We hope this work could encourage text entry researchers to consider word clarity when evaluating smart keyboard, and help improving the empirical practice in this important research field.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Plan under Grant No. 2016YFB1001200, the Natural Science Foundation of China under Grant No. 61272230 and No. 61303076, Tsinghua University Research Funding No. 20151080408.

REFERENCES

1. 2016. A note on calculating text entry speed. (2016). http://www.yorku.ca/mack/RN-TextEntrySpeed.html.
2. 2016. Damn You Auto Correct. (2016). http://www.damnyouautocorrect.com/.
3. 2016. Pareto-Optimized phrase sets. (2016). http://pi.cs.tsinghua.edu.cn/lab/PhraseSets.zip.
4. 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

5. Xiaojun Bi, Shiri Azenkot, Kurt Partridge, and Shumin Zhai. 2013a. Octopus: Evaluating Touchscreen Keyboard Correction and Recognition Algorithms via. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 543–552. DOI: http://dx.doi.org/10.1145/2470654.2470732
6. Xiaojun Bi, Yang Li, and Shumin Zhai. 2013b. 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
7. Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2014. Both Complete and Correct?: Multi-objective Optimization of Touchscreen Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2297–2306. DOI: http://dx.doi.org/10.1145/2556288.2557414
8. Xiaojun Bi, Barton A. Smith, and Shumin Zhai. 2010. Quasi-qwerty Soft Keyboard Optimization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 283–286. DOI: http://dx.doi.org/10.1145/1753326.1753367
9. James Clawson, Thad Starner, Daniel Kohlsdorf, David P. Quigley, and Scott Gilliland. 2014. Texting While Walking: An Evaluation of Mini-qwerty Text Input While On-the-go. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 339–348. DOI: http://dx.doi.org/10.1145/2628363.2628408
10. Mark Dunlop and John Levine. 2012. Multidimensional Pareto Optimization of Touchscreen Keyboards for Speed, Familiarity and Improved Spell Checking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2669–2678. DOI: http://dx.doi.org/10.1145/2207676.2208659
11. Leah Findlater and Jacob Wobbrock. 2012. Personalized Input: Improving Ten-finger Touchscreen Typing Through Automatic Adaptation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 815–824. DOI:http://dx.doi.org/10.1145/2207676.2208520
12. Paul M Fitts. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology* 47, 6 (1954), 381.
13. 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. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 649–658. DOI: http://dx.doi.org/10.1145/2702123.2702503
14. Mayank Goel, Leah Findlater, and Jacob Wobbrock. 2012. WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2687–2696. DOI: http://dx.doi.org/10.1145/2207676.2208662
15. Mayank Goel, Alex Jansen, Travis Mandel, Shwetak N. Patel, and Jacob O. Wobbrock. 2013. ContextType: Using Hand Posture Information to Improve Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2795–2798. DOI: http://dx.doi.org/10.1145/2470654.2481386
16. Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language Modeling for Soft Keyboards. In *Proceedings of the 7th International Conference on Intelligent User Interfaces (IUI '02)*. ACM, New York, NY, USA, 194–195. DOI: http://dx.doi.org/10.1145/502716.502753
17. Poika Isokoski and Roope Raisamo. 2000. Device Independent Text Input: A Rationale and an Example. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '00)*. ACM, New York, NY, USA, 76–83. DOI:http://dx.doi.org/10.1145/345513.345262
18. Clare-Marie Karat, Christine Halverson, Daniel Horn, and John Karat. 1999. Patterns of Entry and Correction in Large Vocabulary Continuous Speech Recognition Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 568–575. DOI: http://dx.doi.org/10.1145/302979.303160
19. Bryan Klimt and Yiming Yang. 2004. Introducing the Enron Corpus.. In *CEAS*.
20. Per Ola Kristensson and Keith Vertanen. 2012. Performance Comparisons of Phrase Sets and Presentation Styles for Text Entry Evaluations. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces (IUI '12)*. ACM, New York, NY, USA, 29–32. DOI: http://dx.doi.org/10.1145/2166966.2166972
21. Per-Ola Kristensson and Shumin Zhai. 2005. Relaxing Stylus Typing Precision by Geometric Pattern Matching. In *Proceedings of the 10th International Conference on Intelligent User Interfaces (IUI '05)*. ACM, New York, NY, USA, 151–158. DOI: http://dx.doi.org/10.1145/1040830.1040867

22. 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

23. Luis A. Leiva and Germán Sanchis-Trilles. 2014. Representatively Memorable: Sampling the Right Phrase Set to Get the Text Entry Experiment Right. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1709–1712. DOI: http://dx.doi.org/10.1145/2556288.2557024

24. I. Scott MacKenzie and R. William Soukoreff. 2002. A Character-level Error Analysis Technique for Evaluating Text Entry Methods. In *Proceedings of the Second Nordic Conference on Human-computer Interaction (NordiCHI '02)*. ACM, New York, NY, USA, 243–246. DOI: http://dx.doi.org/10.1145/572020.572056

25. 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

26. I Scott MacKenzie, Shawn X Zhang, and R William Soukoreff. 1999. Text entry using soft keyboards. *Behaviour & information technology* 18, 4 (1999), 235–244.

27. Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. 2014. Vulture: A Mid-air Word-gesture Keyboard. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1073–1082. DOI: http://dx.doi.org/10.1145/2556288.2556964

28. Frank J Massey Jr. 1951. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association* 46, 253 (1951), 68–78.

29. Paul Nation and Robert Waring. 1997. Vocabulary size, text coverage and word lists. *Vocabulary: Description, acquisition and pedagogy* 14 (1997), 6–19.

30. Hugo Nicolau and Joaquim Jorge. 2012. Touch Typing Using Thumbs: Understanding the Effect of Mobility and Hand Posture. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2683–2686. DOI: http://dx.doi.org/10.1145/2207676.2208661

31. Tim Paek and Bo-June (Paul) Hsu. 2011. Sampling Representative Phrase Sets for Text Entry Experiments: A Procedure and Public Resource. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2477–2480. DOI: http://dx.doi.org/10.1145/1978942.1979304

32. 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

33. Germán Sanchis-Trilles and Luis A. Leiva. 2014. A Systematic Comparison of 3 Phrase Sampling Methods for Text Entry Experiments in 10 Languages. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 537–542. DOI: http://dx.doi.org/10.1145/2628363.2634229

34. Garth Shoemaker, Leah Findlater, Jessica Q Dawson, and Kellogg S Booth. 2009. Mid-air text input techniques for very large wall displays. In *Proc. GI'09*. Canadian Information Processing Society, 231–238.

35. Katie A Siek, Yvonne Rogers, and Kay H Connelly. 2005. Fat finger worries: how older and younger users physically interact with PDAs. In *IFIP Conference on Human-Computer Interaction*. Springer, 267–280.

36. Miika Silfverberg, I. Scott MacKenzie, and Panu Korhonen. 2000. Predicting Text Entry Speed on Mobile Phones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '00)*. ACM, New York, NY, USA, 9–16. DOI: http://dx.doi.org/10.1145/332040.332044

37. Brian A. Smith, Xiaojun Bi, and Shumin Zhai. 2015. Optimizing Touchscreen Keyboards for Gesture Typing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3365–3374. DOI: http://dx.doi.org/10.1145/2702123.2702357

38. 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

39. 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, New York, NY, USA, 295–298. DOI:http://dx.doi.org/10.1145/2037373.2037418

40. 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

41. Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain Text Entry on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2307–2316. DOI: http://dx.doi.org/10.1145/2556288.2557412

42. Jacob O. Wobbrock and Brad A. Myers. 2006. Analyzing the Input Stream for Character- Level Errors in Unconstrained Text Entry Evaluations. *ACM Trans. Comput.-Hum. Interact.* 13, 4 (Dec. 2006), 458–489. DOI:http://dx.doi.org/10.1145/1188816.1188819

43. Shumin Zhai, Alison Sue, and Johnny Accot. 2002. Movement Model, Hits Distribution and Learning in Virtual Keyboarding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 17–24. DOI: http://dx.doi.org/10.1145/503376.503381