An Approach for Improving Thai Text Entry on Touch Screen Mobile Phones Based on Distance and Statistical Language Model

Siwacha Janpinijrut[1], Cholwich Nattee[1], and Prakasith Kayasith[2]

[1] School of ICT, Sirindhorn International Institute of Technology, Thammasat University, Thailand

[2] National Electronics and Computer Technology Center National Science and Technology Development Agency (NSTDA), Thailand Science Park

siwacha.janpinijrut@student.siit.tu.ac.th
cholwich@siit.tu.ac.th
prakasith.kayasith@nectec.or.th

Abstract. This paper introduces an approach for improving Thai text inputting via virtual keyboard on touch screen mobile phones. We propose a technique for character candidate selection to choose the most proper character sequence. The proposed approach consists of two main parts i.e. candidate character generation based on touch area, and trigram model for candidate selection. The candidate generation is used to acquire more than one character per one touch. We define the area nearby the touch point as a touch area. The area is defined based on the statistics collected from various users. The characters within the touch area are considered as a set of candidate characters. The candidates are weighted based on their distances from the touch point. Then, the character trigram model is applied to select the combination of characters with the highest probability suggested to user. In order to evaluate the performance of the proposed technique, we evaluate the word-level accuracy of the words generated by the proposed technique and the words generated by combining the characters with the shortest distances from the touch point. Our technique yields better performance. It produces the results with the accuracy of 93.29% in character level and 66.67% in word level, while the accuracy is 83.48% in character level and 48.33% in word level when the nearest characters are selected.

Keywords: virtual keyboard, touch screen mobile phone, word selection, candidate selection, distance-based candidate generation, trigram model.

1 Introduction

Nowadays, a number of mobile phones equipped with touch screen is significantly increased. Most of them are equipped with two widely-used types of touch screens i.e. resistive and capacitive touch screens. In the former type, its panel composes of two metallic, electrically conductive layers separated by a narrow gap. When an object presses down, two layers become connected and shot circuit at that point. The controller then calculates resistance so as to know which point is touched. This type allows any kind of object to touch, but it does not allow multi-touch. While the capacitive touch screen panel consists of insulator coats with transparent conductor. When part of human bodies directly touch on the screen, distortions of electrostatic field occurs and its controller will calculate the touch points. This type allows multi-touch, but it restricts only human bodies touch.

Virtual keyboard is an important input method of the touch phones. Focusing on Thai language, it has more alphabets than English. This causes more problems when we display Thai keyboard on the limited touch screen area. Keys are displayed very close to each other. Even if the standard keyboard layout for Thai language splits all 88 letters into two layers, it still has a lot of buttons per layer. Each button becomes a very small area. Especially on capacitive touch screen type, mostly human finger always larger than the button. This makes it hard to accurately touch on the intended position. This problem causes a lot of errors, and decreases efficiency of Thai text entry on mobile phones.

This paper focuses on reducing the typing error. With very close distance between each character position on small keyboard area, users may not be able to touch the exact area of the character that they want to type. However, based on an assumption that user tries to touch on the exact area. Even they miss the character; the wrong position should be close to the right area. This characteristic is applied in our approach that the characters nearby the touch point are selected as the candidate characters. This should provide a high possibility that the right character is one of the candidates. Another assumption in this paper is that most of words that users type are common words widely used in various documents. So, we grade the candidates with a statistical language model training from a Thai corpus in order to find out the best candidate with the highest probability and suggest to user.

We scope only the performance in term of correctness of our algorithm by ignoring the processing time between each keystroke. So, we use mobile phones only for collecting input data. Then, input data are fetched to process on computer.

This paper composes of five sections. Next, we explain the background of this paper and other related works. Distance-based candidate generation and candidate selection are described. Then, Section 5 shows the experiment method and the results. Finally, we conclude the paper and show our future direction in Section 6.

2 Background

Designing new layout for virtual keyboard is one way to improve performance of text entry. For example, Bi et al. [1] and Zhai et al. [9] introduced a new layout for English language while Sarcar et al. [7] proposed new Indian keyboards. The problem of new layout is that it is hard to make most of users become familiar with.

We intend to improve performance of Thai text entry based on the standard keyboard, Thai Kedmanee keyboard layout, the most well-known Thai keyboard layout, rather than create new layout.

There are many researches proposed for improving text input methods on standard layout of each language. Potipiti et al.[6] addressed an approach to reduce the number of keystrokes on physical keyboard. For Thai users that have to type bi-lingual document i.e. English and Thai, they introduced intelligent keyboard with 2 main properties. The first one is a technique for language identification that automatically switches language without pressing language-switch button. The other approach is key prediction without pressing shift key for Thai language. Both functions use character trigram probabilistic model compare probability between English word and Thai word for Language Identification and compare probability between Thai character with shift key and without shift key for Key Prediction. Brewster and Hughes [2] used pressure of finger touch on touch phone to separate between lowercase and uppercase of English language instead of using shift key. Soft press means a lowercase and hard press means an uppercase letter.

Another way to reduce keystroke is to make a short form of words. Shieber and Baker [8] reduced keystrokes of typing in English by abbreviating input word to short form. They abbreviated by dropping all vowels and consecutive duplicate consonants. Kristensson and Zhai [4] used geometric pattern as a model to do word prediction on soft keyboard using stylus. They trained model by drawing lines between letter key center points in lexicon word, and doing the same thing but with input point sequence to create input pattern. Then, match input pattern against legitimate pattern of word to find out what the most match word.

MacKenzie and Zhang [5] introduced eye typing technique in English language. User typed by gazing at on-screen keyboard on computer screen via eye tracking equipment. Fixation algorithm used to determine which button receives eye-over. They considered character button nearby eye-focus position as candidates. Combining with knowledge of previous character and language model, they can predict which button user should type. This technique suitable for low accuracy input method. Trying to touch on button that smaller size than user finger is one kind of low accuracy method. So, this paper applies this technique to typing on touch screen.

3 Distance-Based Candidate Generation

Keyboard buttons on virtual keyboard have smaller size than users finger so it is hard to touch on the right position. But, characters that users want to type should nearby their touch point. Bases on this hypothesis, we consider all characters close to touch point as candidates. With too small area, it may not cover the right character that user wants to type. Whereas, with too large area, it gets a lot of candidate characters that affect the processing performance with insufficient resources in mobile phone devices. Although this paper does not consider processing time as the factor yet. But, in our future work, we plan to implement our algorithm running on device. Suitable touch area should be the area that most people touch on. So, we find out which area those users mostly touch on each character. We use the distribution of touch points to identify the suitable area of each key. We create character sequence by shuffling all of Thai alphabets and some special characters that exist on standard Thai keyboard. It composes of 88 characters. All the 88 characters are split into two keyboard layers. Each layer consists of 44 characters. Both layers have the same layout therefore it has only 44 unique positions. We let 20 users typing that sequence in their style, no specific hand of fingers that user have to use. After that, we get 40 coordinates of each character position.

Fig. 1 shows standard keyboard layout with distribution of 44 distinct character positions.

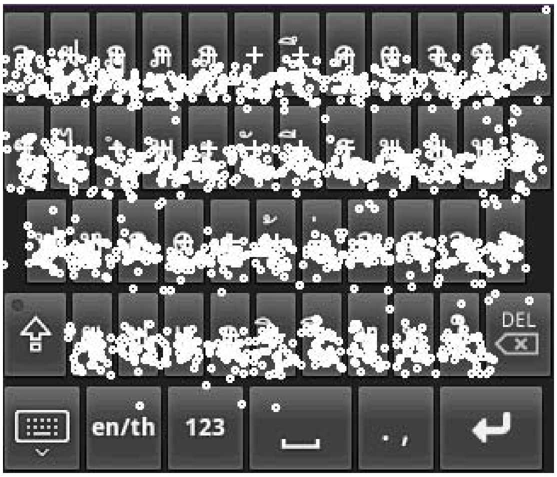


Fig. 1. Distribution of touch point of each character position

3.1 Touch Area

We can find the mean distance and standard deviation for each position.

$$\bar{d_c} = \frac{\sum_{i=1}^{n_c} d_{c_i}}{n_c}, \tag{1}$$

$$SD(d_c) = \frac{\sqrt{\sum_{i=1}^{n_c} (d_{c_i} - \bar{d_c})^2}}{n_c - 1} \tag{2}$$

where
d_{c_i} is distance between touch point and center point position c,
$\bar{d_c}$ is average distance of position c,
n_c is number of touch point of position c.

After we obtain the average distance and SD of each position, we find the average distance and the average SD for all key positions with the following equations.

$$\bar{d} = \frac{\sum_{c=1}^{44} \bar{d_c}}{44}, \tag{3}$$

$$\overline{SD(d)} = \frac{\sum_{c=1}^{44} SD(d_c)}{44} \tag{4}$$

The suitable radius should cover average distance and most of its distribution. Therefore, we set the radius as the following equation.

$$\bar{r} = \bar{d} + 2\overline{SD(d)} \tag{5}$$

However, the radius should not be larger than maximum distance. This equation shows maximum radius it should be.

$$r_{max} = \max_c(\bar{d_c} + 2SD(d_c)) \tag{6}$$

In Section 5, we conduct experiments with various sizes of the radius i.e. 25, 27, 29 ($\bar{r}$), 31, . . . , 43 (r_{max}) to find out the most suitable radius..Furthermore, if we consider the distribution of touch points in x and y axes separately, we will find another interesting thing. The distribution on x axis is not biased to the left or right side for most character position. But, most of the distribution on y axis is biased down side as shown in Table 1.

Table 1. Average radius and SD on y axis reference with center point of each row

Keyboard row	$\bar{d_y}$	$SD(d_y)$
1st	15.15	7.71
2nd	10.64	7.29
3rd	7.03	7.12
4th	7.95	7.64

The top row is the most shifted down while the lower rows are slightly shifted. Based on this distribution, the center point of each key button may not be a good reference point. So, we propose a new reference point by averaging the y values of each row. Then, 16, 11, 7 and 8 pixels are the shift down distance from center point of button of the top to bottom rows, respectively. We call it as center of distribution keyboard layout and call the ordinary layout as center of button keyboard layout.

3.2 Candidate Weight

Each candidate character covered by touch area have a different weight depending on its distance from the touch point.

$$w_c = 1 - \frac{d_c}{r} \tag{7}$$

where
w_c is distance weight of candidate character c,
d_c is distance of character c from touch point,
r is radius of touch area.

This weighting scheme is applied based on the concept that user intends to touch the point as close as possible. The character with smaller distance from the touch point, gains higher weight. This value is used to weight the probability of the character obtained by the language model. The character with lower probability may win the higher one if it has higher distance weight.

4 Candidate Selection

N-gram model is widely-used technique in language processing. The more number of n, the more correctness of statistic model is produced and the more system resources are used to process, too. The high number of n cannot apply in this research because of the limited resource on mobile device. Then, trigram which has n equal to three is selected to create character level language model by obtaining the statistics from a Thai corpus named BEST2010 [3] created by NECTEC. This corpus consists of more than 5 million words.

From a sequence of keystrokes, we generate a set of candidate sequences. Each candidate concatenates all possible candidates to create all possible character sequences. The character sequence that has higher probability than any other sequence is presented to user as the most appropriate word. The probability for each character sequence is calculated based on the character trigram model with the distance weight as shown in the following equation.

$$P(c_1 c_2 ... c_n) = \prod_{i=1}^{n} P(c_i | c_{i-2} c_{i-1}) w_i \tag{8}$$

Where
$P(c_1 c_2 ... c_n)$ is probability of each character sequence,
$P(c_i | c_{i-2} c_{i-1})$ is probability of character c_i follow by character $c_{i-2}c_{i-1}$,
w_i is distance weight of character c_i.

The flowchart of proposed methods is shown in Fig. 2.

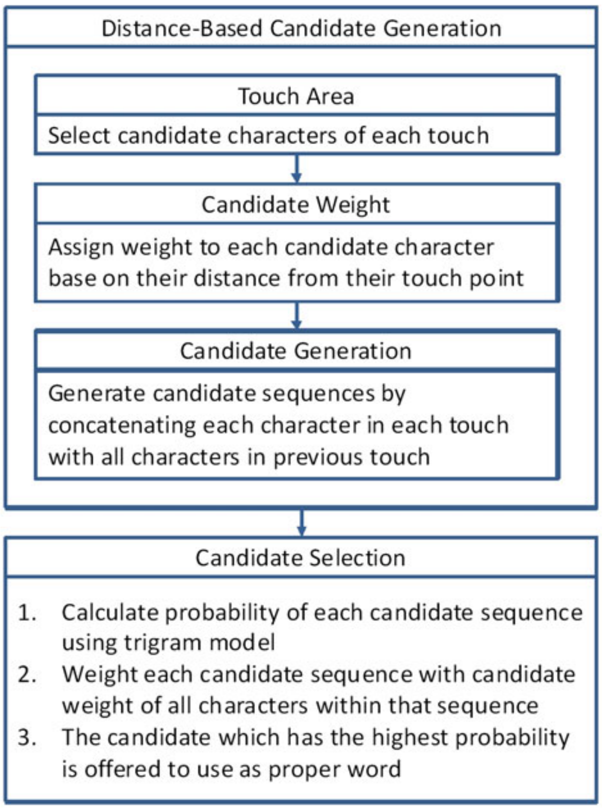

Distance-Based Candidate Generation
Touch Area
Select candidate characters of each touch
Candidate Weight
Assign weight to each candidate character base on their distance from their touch point
Candidate Generation
Generate candidate sequences by concatenating each character in each touch with all characters in previous touch
Candidate Selection
1. Calculate probability of each candidate sequence using trigram model
2. Weight each candidate sequence with candidate weight of all characters within that sequence
3. The candidate which has the highest probability is offered to use as proper word

Fig. 2. Flowchart of proposed methods

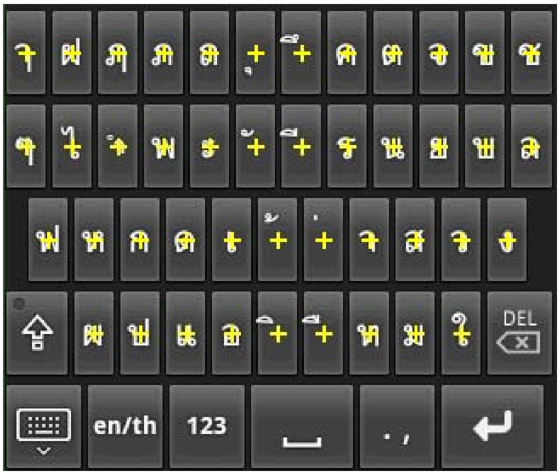
5 Experiments

There are many touch screen phones available in the market. Android phone becomes a hot new trend and is growing up faster. HTC Magic is Android phone that we use in our work. Its resolution is 320 × 480 pixels. Its default operating system is Android 1.6. CN Thai keyboard layout is chosen to be experiment keyboard.

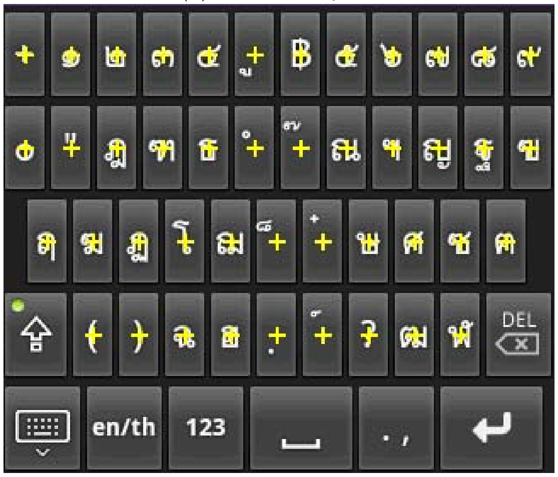
Most applications on touch screen phone allow rotatable; change orientation of use, including text input method. But, we focus only on portrait orientation layout that has smaller keyboard area than landscape orientation.

Each character on keyboard is represented by the center point of its button as shown in Fig. 3. The distance between the center points of two horizontally adjacent characters is 27 pixels. While the distance between the center points of two adjacent rows is 55 pixels.

We randomly choose 80 words and 20 words from the corpus to be a training set and a test set respectively. Both sets consist of words with high, medium



(a) Shift-off layout

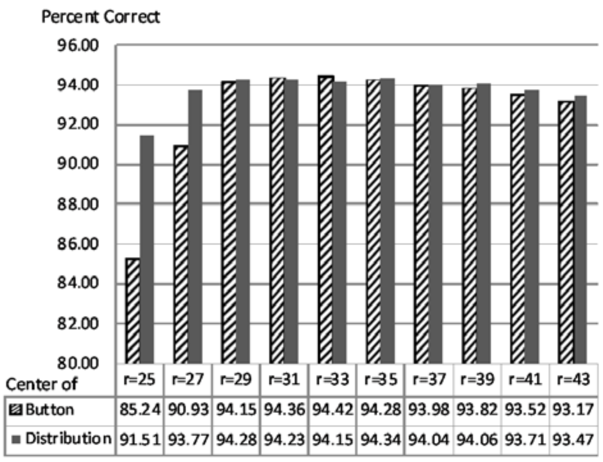


(b) Shift-on layout

Fig. 3. Center point of keyboard button

and low frequency as well as long, medium and short word length in the same average number of each set.

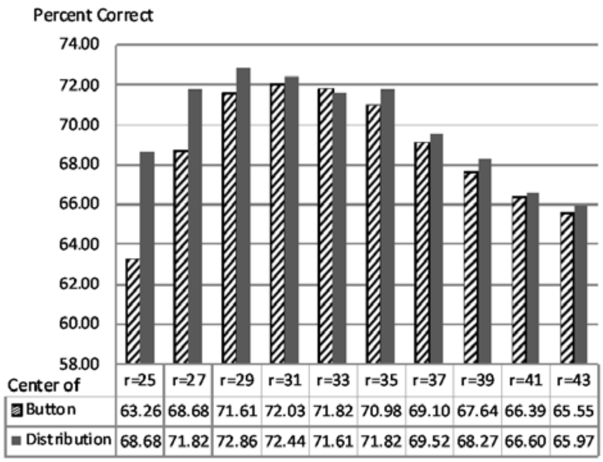
With the training set, we conduct the experiments using various radiuses and two kinds of keyboard layouts mentioned in the previous section. We let six users type all training set words in their style on the mobile phone, word by word, to get 480 coordinate sequences. Every coordinate sequences are processed by our algorithm to get all possible character sequences with its probability. Then, we select the character sequences that have most probability as the output words. We match 480 the obtained words against the correct words to find out the accuracy in character and word levels. In character level, we match and count the number of characters in words, and we match word by word in word level.



Center of	r=25	r=27	r=29	r=31	r=33	r=35	r=37	r=39	r=41	r=43
▨ Button	85.24	90.93	94.15	94.36	94.42	94.28	93.98	93.82	93.52	93.17
■ Distribution	91.51	93.77	94.28	94.23	94.15	94.34	94.04	94.06	93.71	93.47

(a)

Fig. 4. Percent correctness in character level of various radius and layout



Center of	r=25	r=27	r=29	r=31	r=33	r=35	r=37	r=39	r=41	r=43
▨ Button	63.26	68.68	71.61	72.03	71.82	70.98	69.10	67.64	66.39	65.55
■ Distribution	68.68	71.82	72.86	72.44	71.61	71.82	69.52	68.27	66.60	65.97

(b)

Fig. 5. Percent correctness in word level of various radius and layout

Fig. 4 and 5 show the accuracies at the different radiuses in character and word levels respectively.

In character level, the proposed method works well. It yields around 93 – 95% of correctness for most of the radiuses. The performance in word level drops to 63 – 73% since we applied only the character trigram model, it makes sense that its mechanism work well in character level. However, the method did not apply any statistics or information related to words. We plan to apply the language model in word level to improve correctness in the future works.

Table 2. The correctness of different word frequency

Frequency	Method	Character level (%)	Word level(%)
Low	Proposed	90.51	47.62
	Ordinary	88.72	52.38
Medium	Proposed	95.08	73.81
	Ordinary	81.97	38.10
High	Proposed	95.46	80.56
	Ordinary	85.35	55.56

Table 3. The correctness of different word length

Length	Method	Character level (%)	Word level(%)
Short	Proposed	81.61	52.38
	Ordinary	89.66	69.05
Medium	Proposed	93.00	66.67
	Ordinary	83.00	40.48
Long	Proposed	97.71	83.33
	Ordinary	85.42	33.33

Considering the center of button and the center of distribution keyboard layers, both yields not much difference in the result. But in overall, the center of distribution is slightly better than the ordinary center.

For the size of radius, the radius that is shorter than $\bar{r}$ yields a quite low accuracy when it is compared to the others. Both figures show that the suitable radius should be between 29 and 35 pixels. We then select the radius of 35 pixels because it has the highest percent of correctness of all radiuses in character level.

Finally, we conduct the experiment on the rest 20 random words in the test set to find out the actual performance of our algorithm against the ordinary typing method. The correctness of the ordinary method in character level is 54.43% and 48.33% in word level. While our algorithm perform 93.29% and 66.67% in character and word level respectively. This result shows that our algorithm provides the improvement in the text entry accuracy and the performance may increase if we introduce the language model in word level.

We also have more detail about these results. We separately consider the test set in two viewpoints i.e. word frequency aspect and word length aspect. These results show in table 2 and 3 respectively.

Words appear in corpus more than 1,000 times, between 100 to 1,000 times and less than 100 times are considered high, medium and low frequency word group, respectively. The proposed method provides more accuracy while the frequency is increase. On the other hand, the frequency does not affect the performance of the ordinary method because it only bases on the distance, it does not use any frequency information.

For the word length aspect, the short word length group has two to six characters per word while the medium word length group has seven to eleven characters per word and the long word length group has twelve to sixteen characters per word. The proposed and ordinary methods both base on the distance so the main different of these two methods is the language model. More character in word makes more information we can obtain from the model thus longer word length yields higher performance of the proposed method on both character and word level.

6 Conclusion and Future Works

This paper uses distance-based candidate generation and character trigram model to reduce error from text input method via virtual keyboard on mobile phones by word selection. We compare two kinds of keyboard layout. One is center of button keyboard layout. The other is center of distribution keyboard layout. The results show that the later layout is better. We found that the best radius is 35 pixels. Our algorithm performs better than the ordinary method about 1.7 times in character level and 1.38 times in word level.

For the future works, we plan to implement our algorithm on a real device in order to evaluate the processing time. We also plan to improve performance of candidate selection by including the language model in the word level into candidate selection method. By introducing the concept of word, it would yield a better performance. Furthermore, we aim to do word completion so as to reduce the number of keystrokes which is the main problem of using Thai keyboard.

In market, there are many brands and models of touch-screen mobile phones. Each model has difference screen size. Thus we plan to apply these proposed methods on various screen size of mobile phone.

Acknowledgments. This research is financially supported by Thailand Advanced Institute of Science and Technology Tokyo Institute of Technology (TAIST&TokyoTech), National Science and Technology Development Agency (NSTDA) Tokyo Institute of Technology (Tokyo Tech) and Sirindhorn International Institute of Technology (SIIT), Thammasat University (TU). The equipment used in this research is supported by the Telecommunication Research and Industrial Development Institute(TRIDI).

References

1. Bi, X., Smith, B., Zhai, S.: Quasi-qwerty soft keyboard optimization. In: CHI 2010: Proceedings of the 28th International Conference on Human Factors in Computing Systems, pp. 283–286. ACM, New York (2010)
2. Brewster, S., Hughes, M.: Pressure-based text entry for mobile devices. In: MobileHCI 2009: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services, pp. 1–4. ACM, New York (2009)
3. Electronics, T..N., (NECTEC), C.T.C.: Benchmark for Enhancing the Standard of Thai language processing 2010 (BEST 2010), `http://www.hlt.nectec.or.th/best/?q=node/10`
4. Kristensson, P., Zhai, S.: Relaxing stylus typing precision by geometric pattern matching. In: IUI 2005: Proceedings of the 10th International Conference on Intelligent User Interfaces, pp. 151–158. ACM, New York (2005)

5. MacKenzie, I.S., Zhang, X.: Eye typing using word and letter prediction and a fixation algorithm. In: ETRA 2008: Proceedings of the 2008 Symposium on Eye Tracking Research & Applications, pp. 55–58. ACM, New York (2008)
6. Potipiti, T., Sornlertlamvanich, V., Thanadkran, K.: Towards an intelligent multilingual keyboard system. In: HLT 2001: Proceedings of the First International Conference on Human Language Technology Research, pp. 1–4. Association for Computational Linguistics, Morristown (2001)
7. Sarcar, S., Ghosh, S., Saha, P., Samanta, D.: Virtual keyboard design: State of the arts and research issues. In: 2010 IEEE, Students' Technology Symposium (TechSym), pp. 289–299 (April 2010)
8. Shieber, S.M., Baker, E.: Abbreviated text input. In: IUI 2003: Proceedings of the 8th International Conference on Intelligent User Interfaces, pp. 293–296. ACM, New York (2003)
9. Zhai, S., Hunter, M., Smith, B.: The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design. In: UIST 2000: Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology, pp. 119–128. ACM, New York (2000)