Pareto Optimized Arabic Mobile Keyboard Layout

Karim El Batran and Mark D Dunlop
University of Strathclyde
Richmond St, Glasgow, G1 1XH, UK
karim.el-batran@strath.ac.uk

Abstract

This paper presents a new design of an Arabic keyboard layout for effective text entry on touch screen mobile phones. Our approach is based on Pareto front optimization using three metrics: minimizing finger travel distance in order to maximize speed, minimizing neighboring key error ambiguities in order to maximize the quality of spell correction, and maximizing familiarity for less technologically literature users through approximate alphabetic sorting. In our short user studies, the new layout showed an observed improvement in typing speed in comparison to a common Arabic layout. We believe the opportunity is now ripe to research new optimized keyboard designs where there is less usage experience than Qwerty has in mainstream Western European languages. Pareto optimisation can produce high quality keyboards for alphabet based laguages that could give real benefits where there is less reluctance to change from Qwerty.

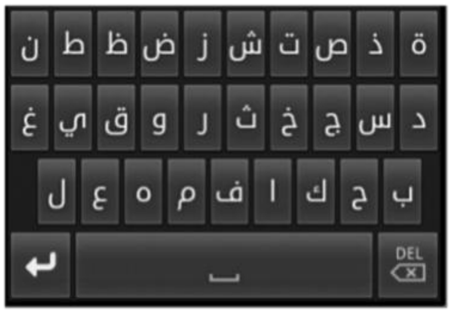

Figure 1. Pareto optimized Arabic keyboard layout

Author Keywords
Touch-screen keyboard design optimization.

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

General Terms
Performance, Design, Human Factors.

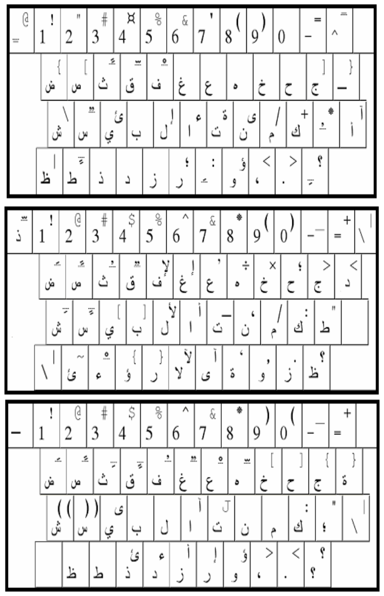

Figure 2. ASMO 663, PC and Apple Arabic layouts

Introduction

There has been a long history of work looking at speed and accuracy in text entry for English [e.g. 9, 12, 16 and 17]. For example, it has long been known that smaller keys can impact on accuracy [14], speed or both [15]. The lack of physical touchable targets in touch-screen smartphones accentuates the "fat finger" problem as users fingers have low accuracy on these screens [7]. Despite Arabic being the seventh most written language (4.23% world population [13]) there has been very little work on Arabic text entry. There is also less keyboard standardization than for English and, at least in   , many users have lower technological backgrounds and are less experienced using a particular keyboard layout than is typical for English. We believe this gives a perfect opportunity to design a keyboard layout that would not suffer the uptake problems of optimized English keyboards.

Optimizing keyboard layouts

For English and many languages the Qwerty layout (and variants such as AZERTY) was adopted on touch screen phones through familiarity. However, touch keyboards have very small keys and no tactile feedback leading to increased error rates on touch screen phones than on physical keyboards [1]. Alternatives keyboard layouts, including Dvorak [5], have not been successful for many reasons, but largely because of the high initial learning curve. Bi, Smith and Zhai introduced an alternative layout to improve Qwerty while maintaining familiarity [4]. They rearranged the Qwerty layout by shuffling keys at most one position from their original location to achieve a quasi-optimized Qwerty variant in the hope of improving acceptance and take-up by users.

Optimization has also been extended to also attempt to reduce the likelihood that hitting a neighboring key will result in a valid dictionary word, and thus improve error correction and accuracy. Dunlop and Levine introduced the concept of badgrams – pairs of letters that when one is substituted for another often result in valid words (e.g. *I* and *O* on a QWERTY keyboard leads to words such as *in/on*, *sin/son*, and *fir/for* being differentiated by very small keyboard distances) [6].

Arabic text entry

The common Arabic desktop keyboard layout is derived from the 1970s Arabic typewriter. However, there is less standardization than for English QWERTY with different keyboard variations in use on different devices in the same country. The Arab Standardization and Metrology Organization (ASMO) developed the standard Arabic desktop keyboard layout supporting the 7-bit Arabic character code [2, 3]. The market however adopted the now widely used Microsoft PC layout, while Apple use an alternative for iOS and OS X. Figure 2 shows these three main variants of the Arabic layout. While not drastically different this will have an impact on users being more open to variations.

There has been no research on optimizing Arabic text entry on mobiles. However, on desktops, Idlebi and Mrayati designed an efficient Arabic keyboard based on statistical approach [10]. Khorshid used a genetic algorithm to optimize the keyboard layout for typing speed. In the genetic algorithm, a fitness function was used taking into consideration key distance, finger used, and hand alternation targeting ten finger typing. Their studies showed a 36.3% speed improvement over the present PC Arabic keyboard [11].

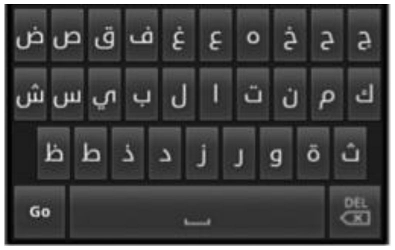

Figure 3. Apple Arabic keyboard layout

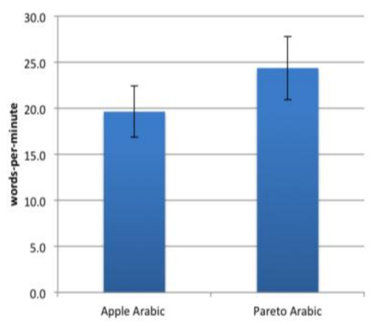

Figure 4. Average estimated words-per-minute

PARETO OPTIMIZATION OF ARABIC

Pareto optimization is a variant of local neighborhood search adapted for multiple criteria searching [8]. We used a three dimensional Pareto front optimization [6] that attempts to optimize for (a) speed of text entry, (b) error correction and (c) familiarity.

The development of a keyboard requires analysis of a large representative text corpus – no such common corpus exists for Arabic. We analyzed 2004 issues of the Arabic newsletter *Watan* containing articles from fields such as culture, economy, news, religion, and sports. The corpus contains 126,913 unique words (34,009,607 occurrences). We calculated bigram and neighbor ambiguity (badgrams) information[1]. The bigram analysis resulted in a weight for each two-letter Arabic bigram – we recorded occurrence information between the 36 characters and between each character and space to give 37*37 pairs. The top four key pair probabilities were ي_ = 0.071, ن_ = 0.068, _ا = 0.060, and ال = 0.054 (where _ represents space). Compared to English there were very few badgrams for Arabic. We believe that this is a result of the reduced role of vowels in written Arabic. The most common badgrams were "ام" and "م يـ", however we only saw 32 badgrams in total (compared to 325 for English), where "ام" represents only 0.003% of the bigram occurrences compared to 0.017% for "AE" in English. A function that measured similarity between novel layouts and Qwerty was included by Dunlop and Levine as a third metric to attempt to minimize deviation from Qwerty. Given the lower levels of computer and keyboard literacy in   together with the variation in desktop layouts, we felt that optimizing to a standard keyboard was less suitable. As such, our third metric was familiarity with a column-major alphabetic keyboard, with the alphabet running from right to left.

An initial set of randomly generated keyboards formed an initial set of potential solutions. This set then went through 2000 iterations in which each keyboard had a small number of keys swapped. If the new keyboard was better on ANY metric than an existing solution, it was added to the set; if it was at least as good on ALL metrics then it *dominates* the existing one, which is discarded. This leads to a *Pareto front* – a set of dominant solutions on a 3D surface. At the end of the process we normalized the space so that each dimension ranged from 0 to 1 and took a good compromise design from the center of the Pareto front.

INITIAL USER EVALUATION

We evaluated our optimized layout against the Apple standard with HTC Desire smartphones (Fig 1,3). We recruited 20 participants aged 20-40 of whom 60% were familiar with touch screens and 85% were PC users. In a balanced study, users were asked to enter 3 phrases as practice then 20 test phrases on each keyboard. Phrases were translations from the English standard MacKenzie and Soukoreff set [18].

RESULTS

The results in figure 4 show that users typed at an estimated 24.4 words-per-minute on average using the Pareto optimized Arabic keyboard layout, but 19.6 wpm using the Apple layout[2]. Despite the relatively short

[1] Bigram and badgram tables, HTML5 demo and phrase sets are available at http://xxx.xxx.xxx/xxx/xx

[2] Our system failed to record wpm data from some phones in our study. Times for these were estimated based on server logs and estimates crosschecked where full data was available.

study this shows an indicative speed advantage for our keyboard (2- tailed paired t-test, n=20, p<0.1) over the standard keyboard for entry by educated users in . Furthermore, the majority of users preferred our layout due primarily to the near alphabetic key distribution. However, some users noted that more time is needed to practice the optimized keyboard layout. We now plan to extend the study to groups with different IT usage experience and education levels.

DISCUSSION

This paper has introduced a new three-dimensionally optimized Arabic mobile keyboard layout. We observed an indicative speed improvement over a standard layout in a short study. Users appreciated the near-alphabetic layout. This is in strong contrast to studied in English where considerable practice is needed to overcome the heritage of Qwerty.

Based on this work we believe there is a strong opportunity to introduce new optimized keyboards for alphabetical laguages in less technically developed nations (e.g. Arabic and Swahili or even English, French, and Portuguese in Africa and Middle East).

Acknowledgements

To be added on non-anonymous version

References

[1] Allen, J. M., McFarlin, L. A. and Green, T. An In-Depth Look into the Text Entry User Experience on the iPhone. In Proc. 52nd HFES (2008), 508-512.

[2] Arab Standardization and Metrology Organization 1985. Standard 449: 7-Bit Arabic Code.

[3] Arab Standardization and Metrology Organization 1987. Standard 663: Arabic Keyboard Layout.

[4] Bi, X., Smith, B. A. and Zhai, S. Quasi-qwerty soft keyboard optimization. *CHI 2010*.

[5] David, P.A. Clio and the Economics of QWERTY. *American Economic Review* 75 (1985).

[6] Dunlop, M. D. and Levine, J. Multidimensional Pareto optimization of touchscreen keyboards for speed, familiarity and improved spell checking, *CHI 2012*.

[7] Holz, C. and Baudisch, P. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. *CHI 2010*.

[8] Hoos, H.H. and Stutzle, T. Stochastic Local Search: Foundations and Applications. Morgan Kaufmann (2005).

[9] Hunter, M., Zhai, S., and Smith, B. A. Physics-based graphical keyboard design. *CHI 2000*.

[10] Idlebi, N. and Mrayati, M. Design Arabic keyboard layout based on statistical properties of Arabic characters, *Computers and the Arabic Language*, 1990.

[11] Khorshid, E., Alfadli, A. and Majeed, M. 'A new optimal Arabic keyboard layout using genetic algorithm', Int. J. Design Engineering, 3(1) (2010).

[12] Lewis, J. R., Kennedy, P. J. and LaLomia, M. J. Development of a Digram-Based Typing Key Layout for Single-Finger/Stylus Input. *HFES 1999*.

[13] http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

[14] MacKenzie, I. S. and Soukoreff, R. W. Theoretical upper and lower bounds on typing speeds using a stylus and soft keyboard. *Behaviour & Info Tech*, 14(6) 1995.

[15] Sears, A., Revis, D., etl al. Investigating touchscreen typing: the effect of keyboard size on typing speed. Behavour & Info Tech 12(1) 1993.

[16] Zhai, S., Hunter, M., & Smith, B. A. The Metropolis keyboard: An exploration of quantitative techniques for virtual keyboard design. UIST 2000.

[17] Zhai, S., Hunter, M., and Smith, B. A. Performance Optimization of Virtual Keyboards. *Journal of Human Computer Interaction*, 17(2 & 3) 2002.

[18] MacKenzie, I. S., & Soukoreff, R. W. Phrase sets for evaluating text entry techniques. *CHI 2003*.