

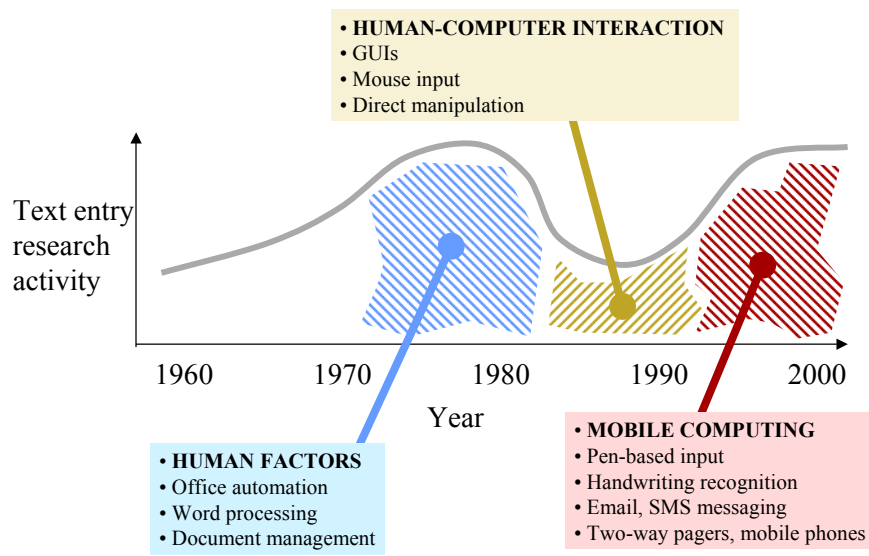
Text Entry Error Analyses

I. Scott MacKenzie

York University
Toronto, Canada



Text Entry Research - Timeline



Mobile Computing



3

Mobile Text Entry

- Tasks
 - Memo pad, address book, calendar entries, to do, travel expenses, email, SMS messaging, etc.
- Devices
 - Stylus
 - Handwriting recognition
 - Tapping on soft keyboard
 - Keyboard
 - 5-button pager
 - 12-key mobile phone
 - Full miniature keyboard (PIMs, some pagers, some mobile phones, new and/or anticipated devices)

4



Text Messaging

- Short Message Service (SMS)
- Phone to phone (maximum 160 characters)
- Part of GSM standard since 1991
- Emerging use in North America
- Widely used in Europe
- About 1 billion messages per day!

5



SMS: Why and How

- Why
 - Cheap (a few cents per message)
 - Asynchronous (like email)
 - Discrete (can send a message during a meeting; while Mom and Dad are sleeping!)
- How
 - Mobile phone keypad (multitap, T9)
 - Qwerty keyboard (Nokia 5510)

6

Text Entry Research

- Two issues collide...
 - Lots of users enter lots of text on lots of mobile devices
 - Mobile text entry techniques are very poor
- So...
 - Lots of researchers are searching for better text entry methods
 - New methods must be tested, compared, etc.
 - How do we evaluate a text entry technique?

7

Evaluation

- Research questions
 - Typically, something like...
 - Is method A as fast/accurate as method B?
 - How much practice to reach, say, 15 wpm?
- Research questions come together in experiments as...
 - Independent variables, and
 - Dependent variables

8

Independent Variables

- These are the **factors** and **levels** in an experiment
- Examples

Factors	Levels
Input technique	Multitap, T9
Keyboard layout	Qwerty, Opti, Fitaly
Key size	small, medium, large
Type of feedback	visual, aural, both, neither
Session	1, 2, 3 ... 10
Word prediction	off, on
Gender	male, female

9

Opti vs. Qwerty

- Two Independent variables
 - **Keyboard layout** with 2 levels: Opti, Qwerty
 - **Session** with 20 levels: 1, 2, 3, ... 20
- Referred to as a **2 x 20 factorial design**
- The 40 test conditions were given to all participants, thus we have a **2 x 20 within-subjects design** (i.e., each subject received all 40 test conditions)
- Note: within-subject design = **repeated measures design** (cf. **between-subjects design**)

10



Dependent Variables

- These are the behaviours measured
- Examples

Variable	Units
Speed	Words per minute (wpm)
Accuracy	Percent errors (%)
Key activity	Keystrokes per character (kspc)
Backspace key events	Count or ratio
"Other" events	Count or ratio

11



Speed as a Dependent Variable

- Relatively straight forward to measure
- Example...

1 2 3 4
1234567890123456789012345678901234567890123
the quick brown fox jumps over the lazy dog

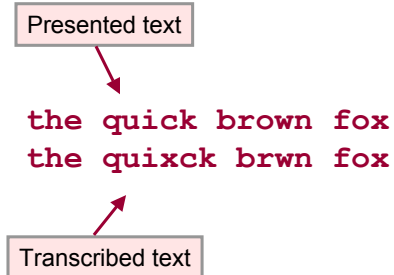
$t = 60$ seconds = 1 minute
Number of characters = 43
Number of words = $43 / 5 = 8.6$
Speed = $8.6 / 1 = 8.6$ wpm

Note: Definition of a **word**: "five characters, including spaces, punctuation, etc"

12

Accuracy as a Dependent Variable

- A bit trickier
- Example...



1. How many errors?

2 (gee, that was easy)

2. What are the errors?

- An "x" was inserted
- An "o" was omitted

3. What is the error rate (%)?

$ER = 2 / 19 = 0.105 = 10.5\%$

13

Minimum String Distance (MSD)

- In the example, there were two errors
- Relatively easy for us to spot
- Hard to automate (i.e., use software)
- Can be done using an algorithm from DNA analysis to compute the minimum distance between two strings
- Consider...

computer
coxzter

14

		-1	0	1	2	3	4	5	6
			c	o	x	z	t	e	r
-1		0	1	2	3	4	5	6	7
0	c	1	0	1	2	3	4	5	6
1	o	2	1	0	1	2	3	4	5
2	m	3	2	1	1	2	3	4	5
3	p	4	3	2	2	2	3	4	5
4	u	5	4	3	3	3	3	4	5
5	t	6	5	4	4	4	3	4	5
6	e	7	6	5	5	5	4	3	4
7	r	8	7	6	6	6	5	4	3

15

Pseudo Code Algorithm¹

```

function msd(A, B) {
    for i = 0 to |A| // |A| = the length of A
        D[i, 0] = i
    for j = 0 to |B|
        D[0, j] = j
    for i = 1 to |A|
        for j = 1 to |B|
            D[i, j] = min (
                D[i-1, j] + 1
                D[i, j-1] + 1
                D[i-1, j-1] + r(A[i], B[j])
            )
    return D[|A|, |B|]
}
function r(a, b) {
    if a = b return 0
    otherwise return 1
}
    
```

¹Soukoreff, R. W., & MacKenzie, I. S. (2001). Measuring errors in text entry tasks: An application of the Levenshtein string distance statistic. *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems -- CHI 2001*, 319-320. New York: ACM.

16



MSD Properties

- Well-defined zero

$$\text{msd}(A, B) = 0, \text{ iff } A = B$$

- Bounded

$$0 \leq \text{msd}(A, B) \leq \max(|A|, |B|) \text{ where } |N| = \text{length of string } N$$

- Commutative

$$\text{msd}(A, B) = \text{msd}(B, A)$$

17



Text Entry Error Rate

$$\text{Error Rate} = \frac{\text{msd}(A, B)}{\max(|A|, |B|)} \times 100\%$$

But, there is a problem... Let's see

18

Accuracy - Revisted

Presented text

quickly
qucehkly

Transcribed text

1. How many errors?

3 (that was a bit tricky)

2. What are the errors?

Hmm, let's see

3. What is the error rate?

$ER = 3 / 8.25 = 0.364 = 36.4\%$

quic--kly
qu-cehkly

quic-kly
qucehkly

qui-ckly
qucehkly

qu-ickly
qucehkly

Optimal Alignments

quic--kly
qu-cehkly

quic-kly
qucehkly

qui-ckly
qucehkly

qu-ickly
qucehkly

- The answer to the question "What are the errors?" shows the set of "optimal alignments" (the set of string pairings with the computed MSD)
- Properties
 1. The set size is often > 1
 2. The alignments are often of different lengths (in the example, 9, 8, 8, 8)
- Because of #2, the error formula needs to be tweaked...

Text Entry Error Rate (new)

$$\text{Error Rate} = \frac{\text{msd}(A, B)}{\overline{S_A}} \times 100\%$$

Where $\overline{S_A}$ is the mean size of the alignments (in the example, 8.25)

OK, it's time for a...

Demo

```

Command Prompt - java MSD -m -a -er
text>java MSD -m -a -er
=====
Minimum String Distance Demo
=====
Enter pairs of strings (^z to exit)
hello
helo
MSD = 1
Error rate (old) = 20.0000%
Error rate (new) = 20.0000%
   h e l o
0  1 2 3 4
h  1 0 1 2 3
e  2 1 0 1 2
l  3 2 1 0 1
l  4 3 2 1 1
o  5 4 3 2 1
Alignments: 2, mean size: 5.0
hello
hel-o

hello
he-lo
-----
    
```

Optimal Alignments - revisited

- Properties (again)
 1. The set size is often > 1
 2. The alignments are often of different lengths (in the example, 9, 8, 8, 8)
- We just dealt with #2
- Implications of #1
 - We don't know what the "user" did
 - This is a problem if we wish to do character-level error analyses

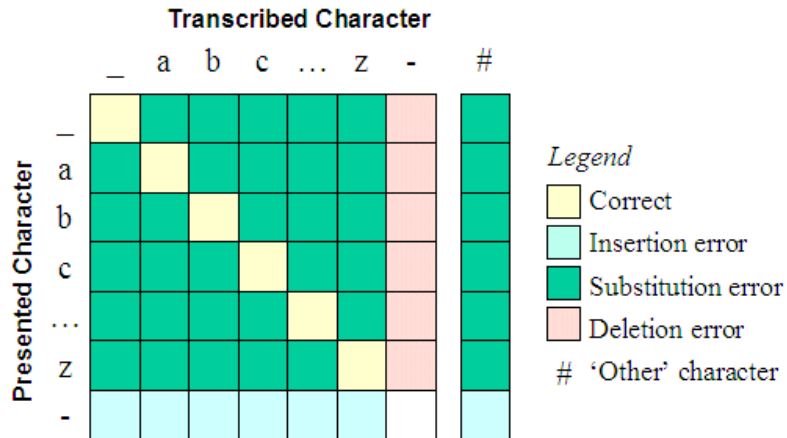
23

Character-Level Error Analyses

- Reasonable compromise
 - Assume each error possibility occurs with equal probability
 - Instead of adding "1" for each error, add "1 / N", where N is the number of alignments, and repeat N times
 - Table view
 - For each character in the alphabet, tally the weighted occurrences of
 - Deletion errors
 - Substitution errors
 - Insertion errors
 - Matrix view (aka *confusion* matrix)
 - When a substitution error occurs, it is often important to know what was substituted (e.g., handwriting recognition)
 - Next slide...

24

Confusion Matrix



25

Demo

```

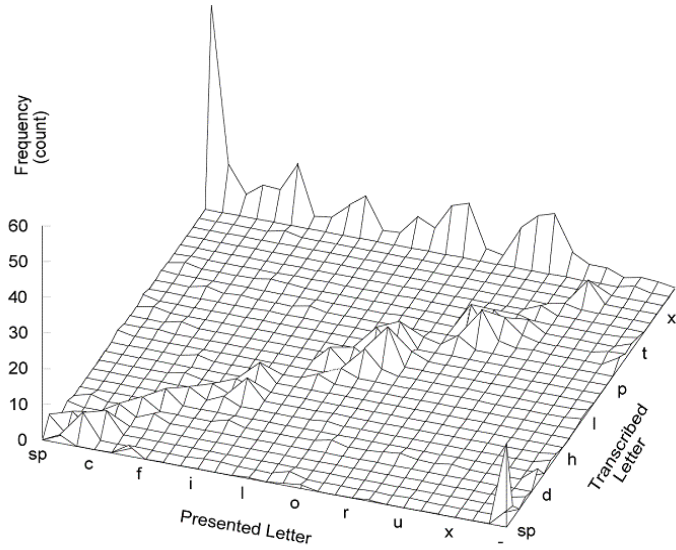
C:\> Command Prompt
text> java ErrorMatrix ds2-phrases.txt
usage: java ErrorMatrix file [-et] [-em] [-a] [-nd] [-pr] [-co]

where file = a file containing presented/transcribed strings
-et = output error table
-em = output error matrix
-a = output alignments (use for debugging/demo)
-nd = null diagonal cell entries in error matrix
-pr = use probabilities instead of counts in error matrix
-co = console output (looks better on display)
(Note: default is no output)

text>
    
```

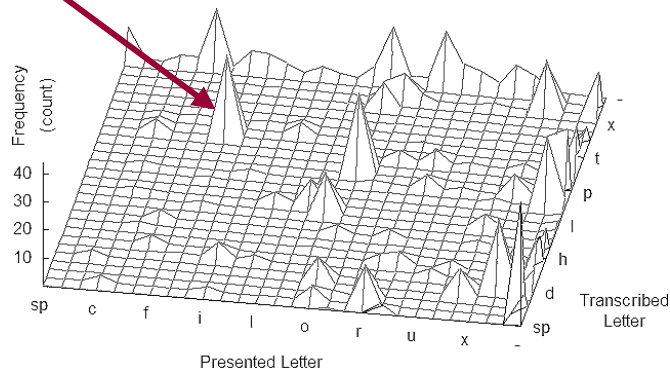
26

Three-Key Text Input Example



Handwriting Recognition Example

'g' often recognized as 'q'



Future Challenges

- In this talk, we dealt only with "presented" vs "transcribed" text
- What about the user's actions that produced the transcribed text?
- The correspondence is inherently one-for-one with a Qwerty keyboard, but this is typically not the case for mobile text entry
- Examples (next slide)

"lazy dog" – Multitap vs T9

Multitap:

lazy dog ← Presented text
 55529999↓999036664 ← Keystrokes
 l az y do g ← Transcribed text

T9:

lazy dog ← Presented text
 5299*0364 ← Keystrokes
 laz y dog ← Transcribed text

Bottom line:

Error analyses must consider the user's input stream, not just the final result.



Thank You

References

- MacKenzie, I. S. (2002). Mobile text entry using three keys. *Proceedings of NordiCHI 2002*, 27-34. New York: ACM.
- MacKenzie, I. S., & Soukoreff, R. W. (2002). Character-level error analyses for evaluating text entry methods, 241-244. New York: ACM.
- MacKenzie, I. S., & Soukoreff, R. W. (2002). Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction*, 147-198.
- MacKenzie, I. S., & Soukoreff, R. W. (2003). Input-based language modeling in the design of high performance input systems. *To appear in Proceedings of Graphics Interface 2003*. Toronto: CIPS.
- MacKenzie, I. S., & Soukoreff, R. W. (2003). Phrase sets for evaluation text entry techniques. *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems -- CHI 2003*, 754-755. New York: ACM.
- Soukoreff, R. W., & MacKenzie, I. S. (2001). Measuring errors in text entry tasks: An application of the Levenshtein string distance statistic. *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems -- CHI 2001*, 319-320. New York: ACM.
- Soukoreff, R. W., & MacKenzie, I. S. (2003). Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric. *Proceedings of the ACM Conference on Human Factors in Computing Systems -- CHI 2003*, 113-120. New York: ACM.