On the application of text input metrics to handwritten text input

Janet C Read

Abstract

This paper describes the current metrics used in text input research, considering those used for discrete text input as well as those used for spoken input. It examines how these metrics might be used for handwritten text input and provides some thoughts about different metrics that might allow for a more fine grained evaluation of recognition improvement or input accuracy.

Text Input Methods

Users spend a significant amount of carrying out text input activities. This time comprises thinking time, input time, and correction time and, more often than not, this time is spent at a keyboard, either a QWERTY keyboard or a reduced keyboard (as found on mobile phones). It is therefore, unsurprising that most of the work on text input has focussed on these two paradigms.

There are several other methods for entering text at a computer, these include gaze typing and spoken input and, with the advent of the PDA and, more recently, the tablet PC, users can use handwritten text that is created with a stylus or pen. This handwritten text can be close to the users 'natural[1]' handwriting, for example, cursive writing, or can require the user to construct letters in a constrained way (as found in the unistroke gestures such as those incorporated in Graffiti) [1], [2].

When the characters that make up a word are entered one by one the text input method can be described as discrete. Apart from in some of the chord keyboards, in keyboard based text input, the text is always entered discretely, in speech applications, and those handwriting applications that allow for 'natural' writing, the text is entered in a continuous stream which makes identification of individual letters problematic.

Handwriting for Text Input

Handwritten text input can be classified across several parameters: As outlined above, it may be discrete or continuous, it may be copied (from text that has already been prepared) or composed. It may also be entered in a constrained way, possibly a character or a word at a time, or may be entered freely (as one would on paper).

Once the text has been created, it may then be being processed or manipulated as digital ink (in this instance it retains its vector format and is not converted into ASCII text) or it may be converted into ASCII text by using recognition software. The recognition software may also vary; it may be a character-wise recogniser, it may include some word level recognition (by referencing a dictionary of possible words) and exceptionally it may also reference a language model allowing for some correction at the sentence level.

[1] The use of the word natural here does not imply that handwriting is a natural activity, rather that the user develops a style of writing that for him or her, feels natural.

The User Experience

In Human Computer Interaction (HCI), it is common to evaluate the user experience with respect to usability parameters including user satisfaction, user-system efficiency and user-system effectiveness [3]. Satisfaction is generally scored by using survey methods and observations, efficiency can be gleaned by recording keystrokes and time on task and on repair, and effectiveness refers to the fit between the task and the technology [4]. In text input activities, the user is generally looking for accuracy and speed whilst looking to have as few errors as possible and wanting those errors that do occur to be easy to spot and easy to fix.

In QWERTY keyboard text input, the optimum performance is one keystroke per character and so recording the number of keystrokes per character by different users allows the performance of individual users at the same, or at different keyboards, to be compared. There are several reduced keyboards that require more than one keystroke per character. These are common on mobile phones and performance at these has been extensively studied by [4].

In some text input activities on reduced keyboards (for instance T9[2]), the user may make all the correct key-presses and still get a wrong representation for the text that they wanted to create. This is due to the predictive nature of the technology which uses probabilistic scoring to determine which word to present in those instances when the combination of key-presses may result in more than one word. An example of this is when the user enters HOME and gets GONE.

When handwriting is used with recognition software, the user may also write the correct letters but the system may produce an incorrect representation of their words [5]. . Unlike the errors using T9 on the mobile phone, these mistakes cannot easily be predicted but in fact, inspection of the writing can often inform the researcher as to why a written word was presented back in a certain way. One difficulty arises when the recognition software refers to a dictionary for a ‘close’ word, at which point, the clues to the mis-recognition that could be gleaned from the character by character recognition can be lost.

The Evaluation of Effectiveness

In much of the text input work in HCI, the effectiveness of the method is measured by making comparisons between two text strings, the first being the presented text (PT), that is what the user was entering, and the second the transcribed text (TT), this being what appears on the screen [6].

For recognition-based interfaces, the term ‘transcribed text’ is a poor description of the final text string as this is in fact text that is generated from the recognition process and, depending on the method of recognition, may vary considerably from the presented text. There are often several other text strings between that which is written and that which is produced, especially when the recognizer is referencing a dictionary having first carried out a character-wise recognition. These intermediate text strings may be very useful in determining how well a user or a system is performing.

The accuracy of the recognition process is typically measured by apportioning a percentage score to text after it has been through the recognition process. The standard metric for this is a percentage error rate.

[2] From Tegic Communications – see www.tegic.com

Percentage Error Measures

As outlined above, the de-facto standard measures for accuracy are generated from two text strings; the presented text (PT) and the Transcribed text (TT). These two strings are compared and each ‘error’ in the generated text is classified as either an insertion (I), a deletion (D) or a substitution (S). This gives a numeric score that is divided by the number or words (or characters) in the prescribed text to give an error rate (E).

This measure can exist in two forms, as a word error rate (as is typically used in speech recognition)

> WER = (S + I + D) / N where N is the total number of words in the test set, and S, I, and D, are the substitutions, insertions and deletions

or as a character error rate (as is typically used in handwriting recognition as well as in discrete text input such as that which is done at a keyboard [7], [8].)

> CER = (S + I + D) / N where N is the total number of characters in the test set, and S, I, and D, are the substitutions, insertions and deletions.

When classifying the errors, penalties or weightings can be applied to the different types of error to ensure that a single substitution is preferred to the combination of a deletion plus an insertion. The American National Institute of Standards and Technology (NIST) uses a weight of four for substitutions and three for deletions and insertions choosing the least weighted score at each error. This is by no means standard and it is more common to score each error at 1 but to choose the error type that is least costly.

To determine a character error rate it is necessary to apply some common sense in lining up the two text strings. To reduce the ‘human; error in this process, work by [6] that applies a minimum string distance algorithm to the two strings is generally used by the text input community. The problems when an MSD algorithm is not used can be seen in the following example:

> Given *quickly* becoming *qucehkly*, application of the MSD algorithm results in an error rate of 3/8 = 37.5%.
>
> Lining up the characters without an MSD would have resulted in an error rate of 6/8 = 75%, given that all the last six letters are in the wrong place.

CER or WER?

As has been suggested earlier, it is common to use the Character Error Rate for handwritten text and the Word Error Rate for spoken text. In part, this distinction is historical as those handwritten text input systems that were first evaluated in this way were typically discrete systems. It does make a difference which metric is calculated. In this following example, the user was required to write the words;

beside the ocean there she sits-

This was then recognized by two different recognisers that initially were set up to recognise at the character level. Table 1 shows how the character error rate is calculated for these two instances.

	Recognition A (character level)	Recognition B (character level)
(TT) CER	renitle the ixean thene yhe sits- (7 + 1 + 0) / 32 = **0.25** (there are 7 substitutions (r for b, n for s, t for d, i for o, x for c, n for r and y for s) and one insertion (l in word 1))	bosiide the occar tneveshe slts- (6 + 1 + 1) / 32 = **0.25** (there are 6 substitutions (o for e, c for e, r for n, n for h, v for r, l for i), one insertion (the i in word 1) and one deletion (of a space))

Table 1 - Character Level Recognition

The same writing was then pushed through the same two recognisers but this time the dictionary was turned on, allowing word level recognition. The results from are shown in the following table with both the character error rate and the word error rate shown:-

	Recognition A (word level)	Recognition B (word level)
(TT) CER	reunite the ixia theme he sits- (7 + 1 + 2) / 32 = **0.31** (there are 7 substitutions (r for b, u for s, t for d, i for o, x for c, i for e, m for r) and one insertion (n in word 1), and two deletions in words 3 and 5.	beside the occur teethe salts- (6 + 1 + 3) / 32 = **0.31** (there are 6 substitutions (c for e, u for a, r for n, e for h, t for s, l for i), one insertion (a in word 5) and three deletions in word 4)
WER	(3 + 0 + 0) / 6 = **0.50** (there are 3 substitutions)	(3 + 1 + 0) / 6 = **0.66** (there are 3 substitutions and one deletion)

Table 2 - Word Level Recognition

It is interesting to see here that the two recognisers scored the same at the character level and that each got worse once the dictionary was used. Interestingly at this point they still scored equally at the character level (although this was just by luck) and that they each scored much worse at the word level, with there then being a difference in performance across the two. This is not always the case; an initial ‘low’ character error rate, once pushed into a dictionary, can sometimes become a zero character and error rate, depending on how well the dictionary fits the language. With high initial character error rates, the dictionary is often unable to improve the character error rates and so high word error rates will result.

Keeping an Eye on the Characters

When a recognition error occurs in handwritten text input, it is most often caused by a relatively poor construction of the character by the user. There are other causes, including software and hardware failure, but these are in the minority [9]. The apparent similarity of some of the characters in the Latin alphabet does little to assist the recognizer in its differentiation. Investigating errors at the 'pre dictionary' character stage, allows the identification of certain simple transformations that can be applied to letters to make them 'transform' into other letters. Four of these transformations are 'grow', 'shrink', 'cut' and 'join'. These transformations account for most of the ambiguity in handwritten text and are increased as writing becomes sloppy (a term used in [10]).
Examples of these are shown in Figure 1.

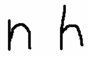
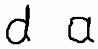
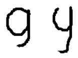
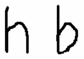
Transform	**Example**
Grow (g)	
Shrink (s)	
Cut (c)	
Join (j)	

Figure 1: Four transformations

These transformations have been seen to correlate with reported mis-recognition pairs. These can be found in work by many authors including [11], [12] and [13]. The single transformations grow, shrink, cut, and join accounted for 94% of the pairs of misrecognition.

Empirical Study

To investigate the transformations, and to consider the appropriateness of CER and WER measures, a small empirical study was carried out with 24 participants. The participants were all teenagers and were roughly matched for gender. They were recruited from a local high school. Each participant copied two common phrases and a selection of three others from nine. The phrases were selected from the list by [14]. The order the phrases were introduced was counterbalanced to reduce any learning effects and the different sets of three from nine were allocated across the sample using a Latin square approach.

The teenagers wrote the phrases onto a tablet PC that was running the Calligrapher recognition software within a custom interface. The interface recorded the writing as an ink trace, the teenagers did not see the recognized text; as far as they were concerned, they were just writing on the screen.

The digital ink was recognized and stored to a text file. Using the presented text (that which was supposed to have been copied) and the transcribed text, following the application of an MSD algorithm, the error rates were calculated at both the word error rate and the character error rate (note that a word level algorithm was used for the word level calculation) and the frequency of the transform substitutions as described in Figure 1 was recorded.

Eight of the twenty four participants used the application with the dictionary on, the other sixteen used the application with the dictionary off. The summary results are shown in Table 3 and Table 4.

	Dictionary on	**Dictionary off**
WER	21%	43%
CER	Nonsense	22%

Table 3 - Character and Word Error Rates

Substitution	**Transform Substitutions**
29%	71%

Table 4 - Ratio of Transform to Non - transform errors

In table 3, it can be seen that there is no percentage for the instance of Character Error Rate with dictionary on. This is because apportioning such a score appears to be nonsensical as by this point there is no longer a mapping between the characters initially for copying and those presented within the words by the recognition software. An example is given here to illustrate this.

> In one instance, the phrase to be copied was: 'Physics and Chemistry are hard' and in one instance, the recognizer (with dictionary support), responded with 'Phases and chemistries we herd'

One often discussed problem with the recognition of handwriting for text input is the two stroke letters. These are known to cause problems, which is one reason why unistroke alphabets like Graffiti fare much better for discrete recognition than 'natural' writing [10]. In this study, the two stroke letters (f, k, t, x) were highly prone to break up.

Other characters that broke up were a, d, g, and q. This was interesting but on investigating these, it can be seen that they can be constructed in two strokes and this natural behavior by the writers may have caused this effect. Interestingly, other possible two stroke letters, b, and p, did not break up, these have the down-stroke first whereas the others have the curve first; this may be the determining factor. The single stroke letters were prone to shrink and stretch, these included c, e, l, m. n. o, r, s u, v, w, y, z. Further work is needed with a larger sample to see if these results can be in any way generalized.

Discussion and Further Work

There are several possibilities for further work from this study. Of particular interest are the non-transform substitutions. Did the user write the wrong character, did he miss out a character, was there a spelling mistake? Another angle is to look at the transform substitutions to see if they can be predicted and/or weighted in order to make either the writing interface learn, or the user improve. One area for further work is to determine whether or not the four transformations should be treated equally. The probability with which people mis-interpret one character as another could be used to determine a weighting for the relevant transformation. The transformations can be extended to incorporate those instances when one character becomes two and when two characters become one. 'a' frequently becomes 'ci' and 'cl' frequently becomes 'd'. There are other regularly occurring situations like this. On first glance, it can be assumed that when 'a' became 'ci' what happened was that 'a' became 'c' and 'i' was inserted. This is not an accurate interpretation; it is extremely rare for users to insert extra characters when doing handwriting. These pseudo-insertions are almost exclusively caused by the splitting of a single character into two by the recogniser. In a similar way, it is unusual for users to fail to write characters; however, consecutive characters being recognised as one letter invariably present themselves as deletions.

It may be possible to define an error measure for handwriting recognition entirely based on substitutions, some of which may replace a single character in the PT with multiple characters in the TT and *vice versa*

This paper has shown how traditional error measures can be applied to handwritten text input. Discussion has focussed on the nature of the recognition errors with some reflection on how knowledge of the nature might inform future work. Further work is needed to validate these ideas.

References

[1] Daly-Jones, O., et al., *Multimodal messages: the pen and voice opportunity.* Interacting with Computers, 1997. **9**: p. 1 - 25.

[2] Lorette, G. *Handwriting recognition or reading? Situation at the dawn of the 3rd Millenium*. in *International Workshop on Frontiers in Handwriting Recognition*. 1998. Taejon. p. 1 - 13.

[3] ISO/IEC, *9241 - 14 Ergonomic requirements for office work with visual display terminals (VDT)s*. 1998.

[4] MacKenzie, I.S. and R.W. Soukoreff, *Text Entry for Mobile Computing: Models and Methods, Theory and Practice.* Human-Computer Interaction, 2002. **17**(2): p. 147 - 198.

[5] Noyes, J., *Talking and writing-how natural in human-machine interaction?* International Journal of Human-Computer Studies, 2001. **55**(4): p. 503-519.

[6] MacKenzie, I.S. and R.W. Soukoreff. *A Character-Level Error Analysis for Evaluating Text Entry Methods*. in *NordiChi2002*. 2002. Aarhus, Denmark: ACM Press. p. 241 - 244.

[7] Read, J.C., S.J. MacFarlane, and C. Casey. *Measuring the Usability of Text Input Methods for Children*. in *HCI2001*. 2001. Lille, France: Springer Verlag. p. 559 - 572.

[8] Mankoff, J. and G. Abowd, *Error Correction Techniques for Handwriting, Speech, and other ambiguous or error prone systems*. 1999, GVU Centre: Georgia, MA.

[9] Read, J.C., S.J. MacFarlane, and C. Casey. *Oops! Silly me! Errors in a Handwriting Recognition-based Text entry Interface for Children*. in *NordiChi 2002*. 2002. Aarhus, Denmark: ACM Press. p. 35 - 40.

[10] Goldberg, D. and D. Richardson. *Touch typing with a stylus*. in *CHI '93*. 1993. Amsterdam. NL: ACM Press. p. 80 - 87.

[11] Frankish, C., R. Hull, and P. Morgan. *Recognition Accuracy and User Acceptance of Pen Interfaces*. in *CHI'95*. 1995. Denver: ACM Press / Addison-Wesley Publishing Co. p. 503 - 510.

[12] MacKenzie, I.S. and L. Chang, *A performance comparison of two handwriting recognizers.* Interacting with Computers, 1999. **11**(3): p. 283 - 297.

[13] Tappert, C.C., C.Y. Suen, and T. Wakahara, *The State of the Art in On-Line Handwriting Recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990. **12**(8): p. 787 - 808.

[14] MacKenzie, I.S. and R.W. Soukoreff. *Phrase Sets for Evaluating Text Entry Techniques*. in *CHI 2003*. 2003. Ft. Lauderdale, FL: ACM Press. p. 754 - 755.