One button text entry

Helen Cousins, supervisor Dr D. MacKay

June 1, 2006

Abstract

Crawler is a one button text entry system designed to help blind or severely disabled people write. Experiments on users indicated that displaying 12 words at a time and using pair correlation made learning to use Crawler easier. Crawler is available to download from www.chiark.greenend.org.uk/ naath/.

1 Introduction

Crawler is a one button text entry system that works by scrolling through a list of words. The user can select a word by clicking the button whilst the word is highlighted. When the user selects a word Crawler then allows the user to reselect it to 'write' it. Because there are more English words than can sensibly be displayed at once Crawler selects a small number of words to display, when a word is selected as well as allowing the user to write that word Crawler also displays more words which the user can then select, these words are selected from those between the word just scrolled past and the word just selected. For instance if the user scrolls past 'apple' and selects 'banana' Crawler then allows the user to enter 'banana' and also displays words between 'apple' and 'banana' which it then scrolls through. There are roughly 6 (depending on the settings used) layers of words covering approximately 90,000 words. Crawler can be trained on any text that the user wants - this training causes more probable words to be faster to find.

The aim was to make Crawler as easy to use as possible - there are several aspects to this. One is making the number of presses required as small as possible because this is a more difficult task than waiting whilst the programme scrolls through words. Also it should be simple to use and easy to learn.

Crawler is inspired by Dr D. MacKay's (and others') work on Dasher which is a mouse based text entry system which uses arithmetic coding to decide which letters the user is more likely to want to enter next. Crawler is a much slower text entry system than Dasher - hence the name.

2 How Crawler works

2.1 Training Crawler

In order to minimise the number of presses needed (and the time taken to find words) more popular words are presented first. This means that first a dictionary needs to be made and words scored according to the probability of their use.

Crawler has four sorts of scoring. Firstly it takes the contents of /usr/share/dict/words (which is a fairly comprehensive list of English words) and scores them according to length - this is because shorter words are much more common than longer ones. If the word is more than 13 characters long then it is given a score of one otherwise it is given a score of 2^{14-l} where l is the length of the word. Crawler also applies the same formula to the file 'ukwords' which contains the UK spellings where these differ from American spellings. Secondly Crawler can be pointed at any text file - this file is read, stripped of punctuation and capitalisation and the number of occurences of each word is counted. The score for each word is 1000 times the number of occurences in the training text(s) (the text file 'files' contains the list of text files that are to be used in this way). Thirdly there is a file 'mydict' to which the user can add words and assign a score; the word and score are read directly from this file. The scores from these three methods are added together to give each word an overall score.

Crawler also counts the instances of pairs of words in the training text(s) and can use this to alter the scoring of words based on the words just entered.

2.2 Selecting words to display

Crawler can display any number of words at a time (limited only by the size of the user's screen); experiments (see section 3) suggest that 12 is a usefull number of words to display at any time.

In order to pick m words to display Crawler first finds the $2m$ words with the highest score in the current list and then selects from these base on an exponential distribution of gaps in cumulative score so that there are larger gaps between the early words in the list than between the later words. This is to compensate for the fact that there is a much longer wait to get to words at the end of the alphabet than there is to get to words at the start (although see experiments for reasons that this is not entirely useful). If there are fewer words than are required then all the available words are used. The 11 words closest to the required cumulative scores are dispayed along with the last word in the current list.

When the user selects a word the available word list changes to those words between the word just scrolled past and the word just selected and the process is repeated.

This system of choosing words to display correctly causes the most common words to be displayed first. Of the 100 most common words only 42 are not in the top 2 layers and all of these are in the third layer (with 12 words displayed, trained on 90% of the text of Emma as for the experiments).

2.3 User Interface

Crawler is a command line programme and uses Curses to interact with the user. The user is presented with a screen with the instructions for using Crawler, the words that have already been input by the user and the current position in the word list. Below this there is the list of words that are available to select. These words are highlighted one after the other and the user can press any button to select the word that is currently highlighted.

Selecting a word moves that word to a column to the left of the screen and displays the next layer of words. Selecting the word on the left adds it to the 'sentence' currently being input, selecting a word in the list repeats this process for the next layer of words.

In addition to the list of words available for selection Crawler has a few commands added at the end of the list - on the first level 'Delete last word' (removing a mistaken word from the sentence) and 'Stop' (to close Crawler) and on all other levels 'go up a level I made a mistake' (to return to the previous layer). Also on the first level the 'word' presented on the left is a full stop.

Figure 1 is a screenshot of Crawler in action, I have entered 'it is a' and am about to enter 'truth'.

2.4 User controlable variables

When Crawler is run the user can set 'scroll', 'wait', 'repeat' and 'm'. Scroll is the speed at which Crawler runs through the words in the list; Wait is the amount of time between Crawler displaying the new words and beginning to scroll through them; Repeat is the number of times that Crawler will go through the words in the list before automatically going up a layer; m is the maximum number of words that are displayed at any time.

```
****************************************************************************
This is a one button communication system.  Crawler will present you with a
list of words and some options for controlling it, to select any word or option
press any key on your keyboard while that word, or option, is highlighted.  The
word you have selected will then appear to the left of a new list of words, if
this was the word you wished to enter select it whilst it is highlighted.  If
the word you want is not in the list that you have select the first word that
is alphabetically after the word you want and then continue with the new list
******************************************************************************

current sentence is:
it is a

You now want a word between
trued and try

*****************************************************************
 trying                 truer
                        trump
                        trust
                        trusted
                        trustworthy
                        trusty
                        truth
                        truth's
                        truthfulness
                        truths
                        try
                        GO UP A LAYER, I MADE A MISTAKE

****************************************************************
```

Figure 1: a screenshot of Crawler in action

A seperate programe (pairwordcounter.py) is used to train Crawler and only needs to be run if the training text has changed.

2.5 Pair correlation

Crawler has the option to use pair correlation in deciding the score of a word. When counting words in the training text Crawler also counts pairs of words. These pair counts are then added to the score of the currently available words. Crawler uses the Hierachical Dirichlet model (D. MacKay and L. Peto 'A Hierarchical Dirichlet Language Model') to do this.

$$P(i|j) = \frac{f_{i|j} + \alpha m_i}{\Sigma(f_{i|j} + \alpha m_i)}$$

Where j=previous word, i=this word, m_i is the probability of i before including the pair correlation and α is a variable parameter (Crawler uses 3) which controlls how important the pair correlation is.

This means that for all words after the first entered the scores of the words are influenced by the previous word entered, such that words more commonly used after the previous word are now easier to get to.

2.6 Example

When Crawler is started the available word list includes all the words that it has been trained on (from 'a' to 'zygote' when trained on /usr/share/dict/words and Emma). The 24 most common words are (in alphabetical order) a, and, as, be, but, for, had, have, he, her, i, in, is, it, not, of, she, that, the, to, very, was, with and you.

The cumulative probability of a word is defined to be the sum of the probabilities of all the words before it (including the word itself). Figure 2 shows the cumulative probabilities of these words. Also marked on figure 2 are the desired locations on the cumulative probability axis of the words to be displayed - these desired probabilities give exponentially decreasing gaps between words.

From these 24 words 11 are chosen by choosing the words closest to the desired probabilities. The chosen words are as, but, for, her, it, not, of, she, the, very and you. To this list is added zygotes which is the last word in the dictionary.

Figures 3 4 and 5 show the cumulative probabilities for an exponential decrease in probability gap between words for the top layer (all words) and two examples from the 2nd layer. These figures show that Crawler is chosing words at roughly the correct cumulative frequencies.

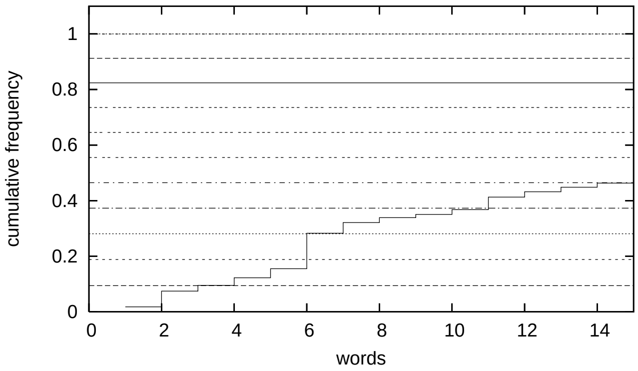

Figure 2: Cumulative frequency of the 24 most common words

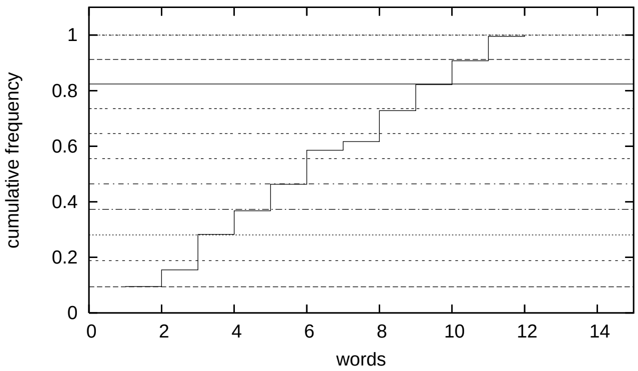

Figure 3: Cumulative frequency of the 12 words of the top layer

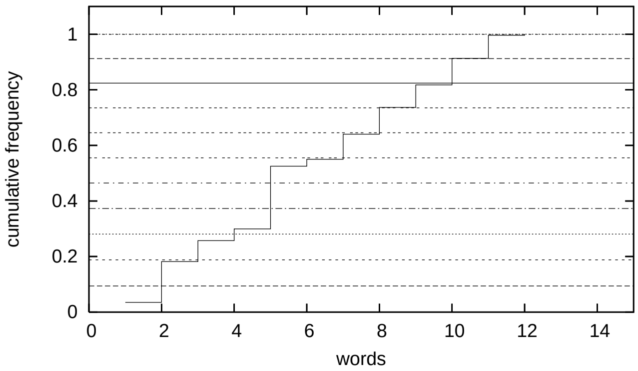

Figure 4: C.F. of one set of words from the second layer

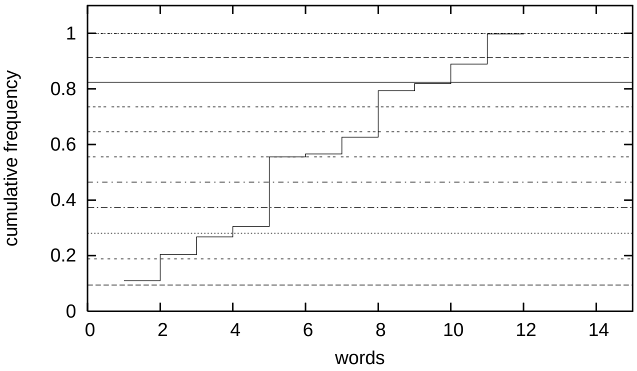

Figure 5: C.F. of another set of words from the second layer

3 Experimenting on users

In order to discover whether Crawler was usable and what settings (number of words displayed, use of pair correlation) were best to use I performed some experiments on three people (A, B and C). There were two main aims - firstly to decide whether $m = 20$ or $m = 12$ was easiest to learn to use and secondly to decide whether using pair correlations made Crawler harder to learn.

Crawler was trained on 90% of the text of Jane Austin's Emma, the remaining 10% was used as dictation for the experiment. Subjects were asked to take dictation using Crawler for 5 minutes at a time. Each subject did two experiments a day for 3 consecutive days and then one final experiment on the following day.

Crawler recorded what each user 'typed' in each experiment so I could calculate the typing speed of the user. The number of 'words' the user entered is taken to be $\frac{n}{5}$ (where n is the number of letters they 'typed') when calculating the users speed in words-per-minute.

For all the experiments the number of repeats was set to 3 and the wait time was set to 1s. The scroll time was changed with each experiment in the same way for all three users. The settings used were 0.8, 0.7, 0.6, 0.5, 0.45 and 0.4 seconds (this is the amount of time Crawler waits on each word before moving to the next).

User A was presented alternately with $m = 20$ and $m = 12$ with pair correlation turned off. Figure 6 shows how user A's 'typing' speed improved. User A improved much faster with $m = 12$, so I concluded that Crawler is easier to learn with $m = 12$. From observing all three users I believe this to be because users frequently missed the word they wanted and needed to wait for Crawler to scroll back to it. A shorter word list meant that the time

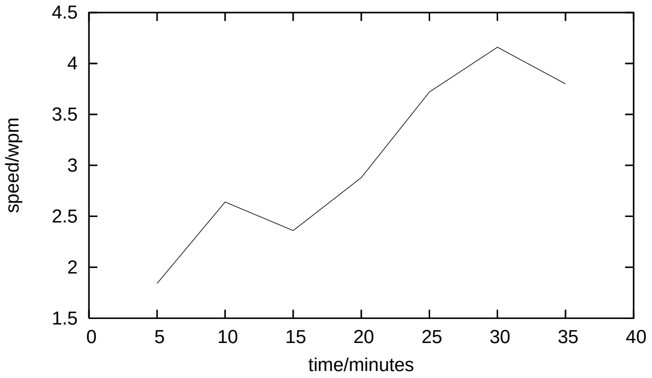

Figure 6: Typing speed of user A

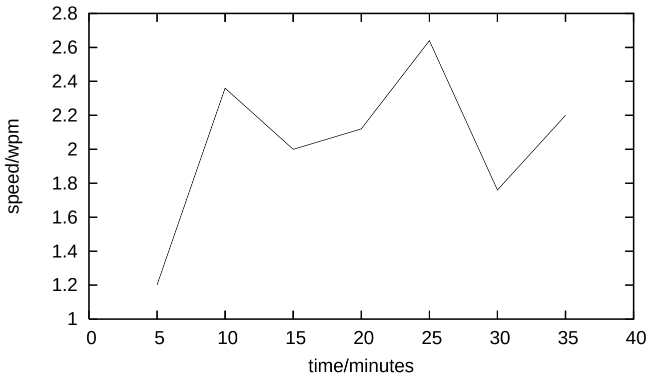

Figure 7: Typing speed of user B

taken to scroll all the way arround was shorter.

Users B and C were presented alternately with pair correlation on and pair correlation off, user B starting with pair correlation on and C with it off. Using the pair correlation reduces the number of presses required for each word and thus should make Crawler faster to use. However if the users were learning by remembering how to reach words that they used often, then changing the words presented by applying pair correlation could possibly cause users to make more errors and thus be slower.

Figures 7 and 8 show how the users improved with each experiment - these users improved (on average) slightly more when the pair correlation was on than when it was off. Additionally neither of the users noticed whether pair correlation was turned on (even though Crawler displays a 'pairs on' label at the bottom) until I pointed it out to them. This suggests that the users were not learning the location of words in the tree.

I also monitered the number of mistakes users made (the number of times

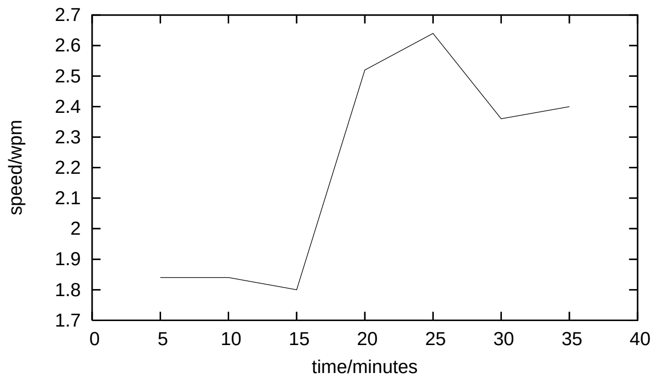

Figure 8: Typing speed of user C

that they selected an incorrect word). Users made on average 4 mistakes in each 5 minute experiment and did not appear to make fewer mistakes as they became faster.

All users felt that the first word in the list was the hardest to select, even with the 'wait' time allowing them to look at the list before scrolling starts. This meant that users often allowed Crawler to scroll through the whole list when they wanted to select the first word. This problem means that it is not especially useful to have larger gaps in cumulative frequency at the start of the presented list (a feature designed so that the user was more likely to want to select words near the top of the list and less likely to need to wait untill Crawler scrolled to the last word).

3.1 User Comments

User A is the least familiar with computers and the least competent typist, users B and C are much faster typists and as such were much more frustrated by Crawler's slowness compared to their usual typing speeds.

All three users commented that the 'go up a layer' option ought to be at the top of the word list rather than at the bottom as they wanted to use it often. This frequent error might reduce with long term use, and experienced users might want to have the option at the bottom of the list.

All three users made more errors because of difficulty with working out where the word they were looking for would be than the did because they selected the wrong word having decided which was the correct word. User A suggested that displaying the alphabet at the top of the screen would be helpful.

4 Possible improvements

There are some improvements that I need to make to Crawler before it can be considered a finished product; some of these are user interface issues that I think would make Crawler an easier programme to use overall and some arise from the suggestions of users.

In order to make Crawler useful for blind people I would like it to be able to read out the words (rather than simply highlight them on a screen). This should be simple to accomplish as programmes to read text aloud already exist. In addition to having Crawler read the words as it scrolls through them it would also need to read out the text that the user had so far entered at suitable intervals. I think that with this alteration Crawler would be reasonably easy for blind people to use although I have not tested Crawler on any blind people.

Crawler currently requires several set up steps that need to be done by someone other than the intended user (assuming that the user requires a one button text entry system). For instance the file 'files' must be edited to include all the training texts and the programme 'pairwordcounter.py' must be run to accomplish the training. Also Crawler's user controlled variables are set on the command line when Crawler is run - this is not the most user friendly method. Ideally Crawler would be entirely operable with a single button interface. This could be done simply by having a menu containing these options, which would be operated in a similar way to text entry when Crawler is started.

It might also be useful to have a method of entering words one letter at a time for when the word that is wanted is not currently in the dictionary. It would be especially useful if Crawler then offered the option to add this word to 'mydict'.

Crawler currently does not train itself during use. Using text produced by the user to further train Crawler during use would make it more adaptive to individual user's writing style.

From the user comments I think that it would be usefull to add the alphabet at the top of the screen. This would not take up too much space and would help at least some users use it faster. It might also be useful to put the 'go up a layer' option first, although this might be less useful for experienced users. These adaptions could be presented as options that the user could set depending on their experience with using Crawler.