Evaluation of an Input Technique for Text Entry using Autosuggestion and -completion

Jonas Puchinger

University of Regensburg

jonas.puchinger@stud.uni-regensburg.de

Maximilian Kögl

University of Regensburg

maximilian.koegl@stud.uni-regensburg.de

Abstract

We implemented and evaluated a form of assisted text entry, where users are given suggestions for completed versions of the word they are currently typing. When compared to unenhanced typing, this method proves to be faster and more error resistant. This allows users to have an overall better experience when writing text on a computer.

Motivation

While some people are really fast with a keyboard when writing text on a computer or similar digital device, others, who may not type as much or have never learned an efficient way of typing are much slower. Another problem when typing without the help of an assisting text input method is often the high error rate. You can be a fast typist, but when you make a lot of errors when typing that fast, your overall time taken for text entry still is not as low as it could be. This is were assisted typing comes in. Many people already use the feature of autocorrection and autosuggestion when typing on their smartphone or tablet, and text editors like word provide similar features. We prove that assisted typing is faster and more error resistant than traditional, unassisted typing by implementing a text editor, that uses the feature of autosuggestion and -completion, and comparing typing performance of this editor to an unenhanced one.

Design and Implementation of the Assisted Typing Method

As already discussed, we chose to implement autosuggestion and autocompletion as the feature to help users type fast and with fewer errors, and thereby providing an overall more efficient typing experience. The method is implemented using Python 3.7 and the PyQt framework. To enable participants to make text entry on a computer, we built a simple text editor with PyQt, based mostly on Qt´s QTextEdit. This editor by itself provides only the most basic features for text entry. To enable participants to experience the features of autosuggestion and autocompletion when using the text editor, we extended its basic version. PyQt has a built-in component that makes it easy to implement autosuggestion and -completion for text entry. This component is called QCompleter. It attaches to a text entry field, in our case the simple text editor based on QTextEdit. Word suggestions are provided to the completion-component by a model, which in our case is a simple list of words in the form of an array, extracted from the sentences we ask the participants of the study to type. With the feature enabled, users can type normally, and are provided with suggestions for complete words from the defined wordlist, that match the sequence of characters they already have put in. These suggestions are shown in a pop-up box just below the text-cursor, can be selected by using the arrow keys, and used for autocompletion by pressing the tab key.

Design of the study

To evaluate our input technique, we conducted a user study, in which we compared the performance of the unenhanced text editor to the one with the assisting feature implemented. The independent variable for this study was whether the assisting typing method was used when typing the provided sentences or not. The dependent variables were the typing speed and the error rate when typing. Our hypothesis before the experiment was that typing speed goes up and the error rate goes down when the assisted typing method is used. We used a sentence generator for the German language (https://satzgenerator.de/) to generate 20 German sentences. These sentences are:

1.	"Korbinian malt sehr gerne mit Wassermalfarben",
2.	"Ein Kiwi kann nicht fliegen",
3.	"Monegassen wohnen in Monaco",
4.	"Die Bundeskanzlerin von Deutschland ist Angela Merkel",
5.	"Viele Menschen mögen eisgekühlten Bommerlunder",
6.	"Mit dem Dietrich knackte der Dieb das Schloss",
7.	"Eine Fachassistentin liebt Juliana auf einer Insel",
8.	"Violett ist eine laute Farbe",
9.	"Der schlagfertige Levin qualmt feige im Schwimmbad",
10.	"Der affige Oleg hat eine orale Lippe",
11.	"Verlogen kommt der Partner mit",
12.	"Je falscher desto selbstloser",
13.	"Die buckelige Petra kichert im Auto"
14.	"Leonard schaltet den Ahorn im Knast ab",
15.	"Planlos fällt Karlottas Mitarbeiter im Kaufhaus auf",
16.	"Es brennt durchgeknallt",
17.	"Je fetter desto betroffener",
18.	"Hitler wird beim ersten Date gehasst",
19.	"Im Swinger Club war dein Lehrer lächerlich",
20.	"Eine Behinderte hat eine Verletzung am Kopf"

We decided to use German sentences, because all participants of the study are German, and because we used the same computer throughout the experiment, which had a German keyboard layout. This consistency in using the same technical setup was also the control variable for the study. We split the 20 sentences into two sets. The first set consists of the first ten sentences shown above, the second one of the remaining sentences. This setup has four conditions, as we have two levels of our independent variable, normal typing (N) and typing with completion (C), and two sets of sentences, (1) and (2), to write for the participants. We decided on a within-subjects design for our study, so each participant used both versions of the text editor. To reduce carry-over- and learning-effects, we counterbalanced these conditions for all four participants in the following order:

Participant	Order of Conditions
1	N1 C2
2	B1 N2
3	N2 C1
4	C2 N1

All four recruited participants were male, German students, who use the German keyboard layout. Participants were asked to write the sentences shown at the top of the application into the text input field just below.

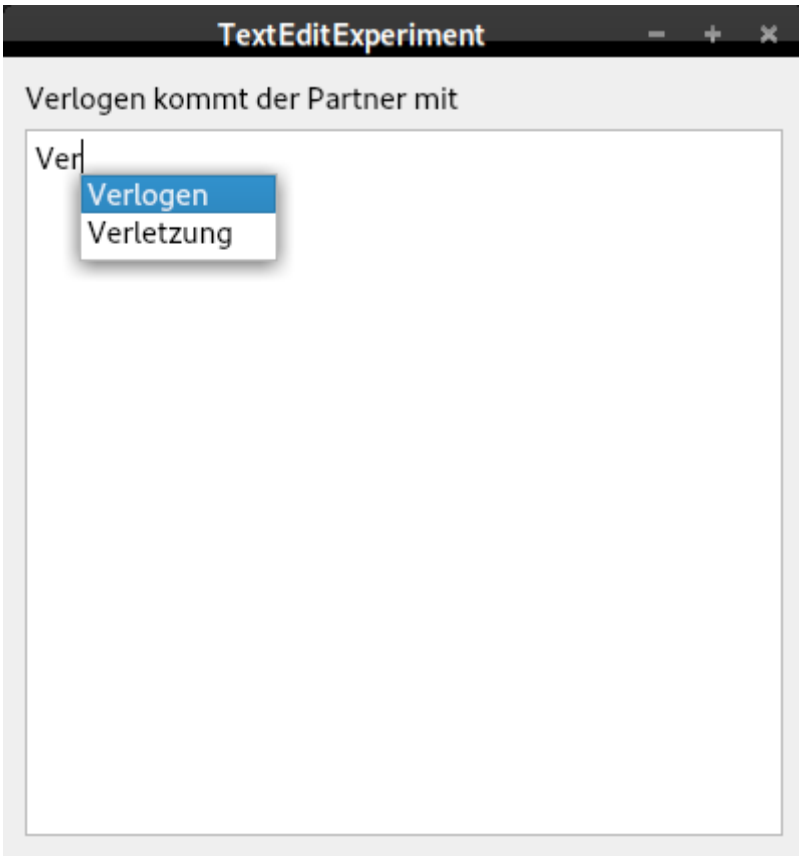

Figure 1: Texteditor with Autocompletion enabled

Results

The measured results for the mean time per word with the completer were 637ms and for the condition without the completer it was 915ms. A t-test showed a significant difference in the completion time for each word in the conditions ($t(3)=-4.97$, $p < 0.0001$).

The measured results for the mean time per sentence with the completer were 6730ms and for the condition, without the completer it was 6618ms. A t-test showed no significant difference in the completion time for each sentence in the conditions ($t(3)=0.22$, $p=0.22$)

Discussion

These results show that with the completion-method it is possible to write single words faster than without the method. But they also show that the needed time to write a sentence is not significantly different than without the method. This raises the question if participants need more time to adjust to the newly completed word or if other aspects slowed the sentence completion time down. Furthermore, we chose only twenty sentences and filled our completer with those words which reduce the number of possible words to complete. Thus it could give other results if the presentation and number of possible completion word are increased.