Reading and Writing
https://doi.org/10.1007/s11145-021-10222-w

Automated extraction of revision events from keystroke data

Rianne Conijn[1] · **Emily Dux Speltz**[2] · **Evgeny Chukharev-Hudilainen**[2]

Accepted: 22 October 2021

Abstract

Revision plays an important role in writing, and as revisions break down the linearity of the writing process, they are crucial in describing writing process dynamics. Keystroke logging and analysis have been used to identify revisions made during writing. Previous approaches include the manual annotation of revisions, building nonlinear S-notations, and the automated extraction of backspace keypresses. However, these approaches are time-intensive, vulnerable to construct, or restricted. Therefore, this article presents a computational approach to the automatic extraction of full revision events from keystroke logs, including both insertions and deletions, as well as the characters typed to replace the deleted text. Within this approach, revision candidates are first automatically extracted, which allows for a simplified manual annotation of revision events. Second, machine learning is used to automatically detect revision events. For this, 7120 revision events were manually annotated in a dataset of keystrokes obtained from 65 students conducting a writing task. The results showed that revision events could be automatically predicted with a relatively high accuracy. In addition, a case study proved that this approach could be easily applied to a new dataset. To conclude, computational approaches can be beneficial in providing automated insights into revisions in writing.

Keywords Revision analysis · Keystroke logging · Eye tracking · Annotation · Machine learning

✉ Rianne Conijn
m.a.conijn@tue.nl

[1] Human-Technology Interaction Group, Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, The Netherlands

[2] Department of English, Iowa State University, Ames, USA

Published online: 22 November 2021

Introduction

Writing involves a variety of processes, ranging from low-level (peripheral) processes, e.g., motor processes such as typing, to high-level (central) processes, such as text evaluation (Olive, 2014). Keystroke analysis has been used to get more insight into these writing processes (Leijten & Van Waes, 2013; Lindgren & Sullivan, 2019). With keystroke analysis, the timestamps for every key pressed and released are recorded and analyzed, giving a detailed overview of the writer's typing behavior. It has been argued that the typing behavior represented in the keystroke logs might provide evidence for these higher-level cognitive processes (Baaijen et al., 2012). For example, a pause before a sentence has been used as an indicator of sentence planning (Medimorec & Risko, 2017; Roeser et al., 2019). In addition, backspace or delete key presses have been related to the revision process (Van Waes et al., 2014).

However, it is still considered difficult to align the metrics obtained from keystroke logging with specific writing processes (Galbraith & Baaijen, 2019). There is a large gap between a single, aggregated metric (e.g., total number of backspaces or mean time between words) and the complex writing processes. One suggested solution to better align the keystrokes with the writing processes has been to combine multiple metrics obtained from the keystroke data (Galbraith & Baaijen, 2019). In the current study we propose a method that combines several metrics to provide a more accurate representation of the writing process, and specifically the revision process.

Revision plays an important role in writing. Although there is no unambiguous relation between revision and text quality, revision has shown to be a determining factor in writers' development, including writers' knowledge about the topic and about writing (Fitzgerald, 1987). Moreover, as revisions lead to disfluencies in the writing process, they are crucial in describing writing process dynamics. Therefore, we specifically focus on how keystroke data can provide insight into revisions made during the writing process.

Revisions have been defined in different ways over the years of writing research, showing a shift in perspective from revisions seen as "error correction", towards the cognitive processes involved in revision (Fitzgerald, 1987). For an overview of the variety of definitions of revisions, see for example (Horning & Becker, 2006; Lindgren & Sullivan, 2006b). In the current study, we employ a rather broad definition of revision to be able to identify a wide range of revisions. Revision is defined as "making any changes at any point in the writing process" (Fitzgerald, 1987, p. 484). This definition focuses on both product and process, and it includes both internal as well as external revisions, where internal revisions are changes in the writer's mind before transcription or text production, while external revisions are changes during the transcription (Murray, 1978). These revisions can be any change in the text, and they do not necessarily have to involve revising an error. In addition, these revisions can include quick, spontaneous corrections (editing), as well as more systematic inspection and improvement (reviewing; Flower & Hayes, 1980). As internal revisions are not visible in

keystroke data, we here only focus on external revisions. External revisions have been further categorized, depending on their spatial location in the text, into pre-contextual revisions and contextual revisions. Pre-contextual revisions are revisions located at the leading edge, while contextual revisions are revisions made within the text produced so far, away from the leading edge (Lindgren & Sullivan, 2006a, 2006b). Pre-contextual revisions are started by a deletion, while contextual revisions can be started by a deletion or an insertion. In the current approach, we focus on both pre-contextual deletions and contextual insertions and deletions.

The current approach has two main advantages compared to previous approaches. First, we consider both deletions and insertions as a possible start of the revision process. Previously, backspace or delete key presses have often been used as an indication of revisions (Van Waes et al., 2014). Within this operationalization only pre-contextual and contextual deletions are considered; contextual insertions are ignored. Second, we do not consider the end of a delete or backspace key as the end of the revision process. A backspace key usually does not represent the full cognitive process of making a revision. For example, a revision often consists of both the deletion of one or more characters, followed by one or more characters that are typed to replace the deleted text (e.g., a substitution of one or more characters). We refer to this full process of making a revision as a '*revision event*'. By considering the full revision event, rather than only the deletion, more information can be obtained about the nature of the revision. For example, natural language processing may be used to automatically identify the scope or the size of the revision. In the current paper we provide a method to extract the full process of each revision event within a keystroke log.

Related work

Although revisions are usually identified by a backspace or delete key press (or sequences of these key presses), some studies already looked at how the full process of a revision, the revision event, could be identified with keystroke data. In the following, we describe each of these approaches, their advantages and disadvantages, and the differences from the current approach.

Events

First, revisions can be seen as an interruption of the text production. Baaijen and colleagues define these interruptions as *events*: "episodes that include other material or operations before the continuation of text production" (Baaijen et al., 2012, p. 256). These operations can include mouse movements and scrolling, as well as contextual insertions and insertions away from the leading edge. This is partly in line with our definition, as the definition includes both deletions as well as insertions.

However, there are two main differences with our current approach. First, mouse movements and scrolling are included in Baaijen and colleagues (2012), as these might indicate rereading and evaluating previously written text. However, this does

not always result in an external revision event. Sometimes the writer might consider that the previously written text does not require any changes, or the writer might use the rereading to plan their next sentence. Accordingly, we only consider revision events that resulted in at least one character being inserted or deleted.

Second, Baaijen and colleagues (2012) only include the operations before the continuation of text production. However, as we argued above, we consider that (part of) the following text production should also be considered as part of the revision event, as this text production might be used to replace the deleted text.

Revision and insertion bursts

Another concept that has often been used to refer to describe interruptions of text productions is *bursts*. The concept of bursts originates from studies on handwriting (*written language bursts*), referring to periods of uninterrupted text production. Specifically, a burst is defined as text production terminated by pauses longer than two seconds (*P-burst*), or text production terminated in an evaluation, revision, or a grammatical discontinuity (*R-burst*; Kaufer et al., 1986).

In typing, bursts are often referred to as sequences of text production bound by interruptions and *pauses*: interkeystroke intervals longer than two seconds. In addition, the type of burst is not only defined based on the nature of the interruption at the end of the burst, but also based on the nature of the interruption at the start of the burst. Baaijen and Galbraith (2018) identify three types of bursts: *P-burst*, bursts that both start and end with a pause; *R-bursts*, bursts that end with a revision at the leading edge, and *I-bursts*, bursts that start with a mouse or arrow-key movement away from the leading edge, followed by text production. The R-burst and I-burst are especially related to our current conceptualization of a revision event, as the R-burst can be seen as a pre-contextual deletion, and the I-burst as contextual insertions.

In the current approach we combine R-bursts and I-bursts into one set of revision events consisting of both insertions and deletions (which can later be further categorized), where each revision event starts with a revision (rather than ends in a revision, as in an R-burst). In this way, we can identify the content of the revision, for both insertions and deletions.

S-notation

A third approach to identify revisions within text production is the S-notation (Eklundh & Kollberg, 2003; Kollberg, 1996; Severinson–Eklundh & Kollberg, 2001). In line with our approach, the S-notation focuses on both insertions and deletions. Specifically, the S-notation transforms the entire writing process to a numbered representation of all insertions and deletions, providing a structured overview of the spatial and temporal location of all insertions and deletions within the writing process. The sentence below shows an example of the S-notation for the production of the following short text: “This is an example of the S-notation. It shows insertions

and deletions." `This is {an example of}[1] |[2] the S{-}[2]notation. |[1] It [consists of][3] |[3] shows insertions and deletions.`

Here, {…} indicates an insertion, […] a deletion, and | a break, or the spatial location in the writing process where the insertion or deletion with the corresponding number started. The numbers indicate the temporal order of the revisions. Within the example sentence, there are three revisions. First, 'an example of' is inserted after the first sentence is finished. Directly thereafter, a hyphen was added in S-notation. Finally, 'consists of' is replaced by 'shows', right on the spot. This indication of both the spatial and temporal location of the revision event is very useful as it can be used to identify episodes of revisions, such as sequences of revisions in previously written text (e.g., the first and the second revision in the example).

Our analysis of revision events is different in two ways. First, the S-notation provides partial information on when the revision ends. The approach indicates the beginning and end of an insertion (denoted by the opening and closing bracket, respectively), but limited information is provided on the end of a deletion. The start of a new revision (insertion or deletion) may be considered as the end of the previous revision, but this might also include some new text production, or production that is not meant to replace the deleted text. Second, our analysis results in a table format rather than a (hierarchical) string. The table format is more reliable, as S-notation can break with long texts, when the hierarchy becomes too complicated (e.g., revisions embedded within revisions, embedded within revisions, etc.). In addition, researchers are more familiar with the table format, making it easier to use and implement.

Revision analyses in keystroke logging tools

Keystroke logging tools also provide built-in approaches to identify revisions. Most keystroke logging tools solely focus on the identification by a backspace or delete key press (Lindgren & Sullivan, 2006a). Two tools provide some more extensive revision analysis: JEdit and Inputlog (Lindgren & Sullivan, 2006a; Van Waes et al., 2012). Both JEdit and Inputlog include the S-notation (as described above). In addition, Inputlog provides a so-called *revision matrix* (Leijten & Van Waes, 2013). This revision matrix provides a linear representation of all insertions and deletions in a table format, making it relatively easy to implement by researchers. The revision matrix includes a row per deletion, insertion, and normal text production. For each of these types, the matrix lists the content (characters deleted, characters inserted, or characters typed, respectively), the number of edits, start and end time of the action, the duration, the position of the first and last character of the action, and the number of characters and words produced or deleted (Leijten & Van Waes, 2013; Leijten et al., 2019).

The revision matrix provides a clear indication of the start and end of an insertion, as well as the start of a deletion. In this sense, the revision matrix can be considered the most similar approach to our current approach. However, although the revision matrix indicates the text production after the deletion, it does not specify which part of the text production following a deletion should also be considered as

part of the revision event. In addition, the revision matrix in Inputlog is dependent on the position of the first and last character of the action, as provided by Microsoft Word. Especially in longer texts with many non-linear transitions, the position of the characters might be inaccurate by one or two characters. Although this is not a problem for many usages of the revision matrix, this would be a problem for our intended application to use natural language processing, for example to identify the scope of a revision. A displacement of just a single character would break several natural language processing algorithms, as the word cannot be correctly identified anymore. Therefore, the current approach is tool-independent, allowing the researcher to opt for different word-processing tools, based on a trade-off between high ecological validity (as in e.g., Microsoft Word) and high accuracy (as in e.g., Notepad).

Current approach

Within the current approach, we define a *revision event* as an external revision, which starts from a pre-contextual deletion or a contextual deletion or insertion, and ends with normal text production or a new external revision. We consider external pre-contextual insertions as new text production at the leading edge (*cf.* Lindgren & Sullivan, 2006a). The revision event is first approximated using a rule-based algorithm, resulting in a *revision candidate*. Thereafter, human annotators indicate whether the revision candidate is indeed a revision event, and annotate where the revision ends. This rule-based approximation significantly speeds up the manual annotation process, as it helps the annotator directly focus on potential revision events. After the manual annotation, we use the refinements from the manual annotations to automatically extract revision events using machine learning.

Rule-based approximation of revision candidates

Start of the revision candidate. The start of a revision candidate is indicated by a pre-contextual deletion or a contextual insertion or deletion. The start of a pre-contextual or a contextual deletion is identified in the keystroke data when a keypress results in a decrease of the text length. Note that within this approach, it does not matter whether the deletion is caused by pressing the backspace or the delete key, or even by a selection of a text block that is followed by a keypress.

The start of a contextual insertion is slightly harder to define, as we need to distinguish contextual from pre-contextual insertions. We define a pre-contextual insertion as a text insertion at the leading edge of the text. The leading edge is defined as the location in the text where internal concepts and forms are transcribed and put into text (Lindgren et al., 2019). The leading edge is often operationalized as the location after the last character of the text that is produced so far. However, this operationalization fails when there are invisible characters at the end of the text (Lindgren et al., 2019), or when there are words, sentences or even full sections at the end of the text that are not used in the analyzed writing session (e.g., a bibliography section). This is especially a problem

for writing spread out over multiple sessions. Therefore, in the current method, we flag all deletions and all insertions away from the current cursor position (point of inscription) as revision candidates. Then, manual annotation is used to indicate whether this insertion is indeed a true revision. This manual annotation is in turn used to improve the indication of the start of the revision event.

In summary, the start of a revision candidate is approximated using a rule-based algorithm based on two rules. Specifically, a revision candidate starts if one of the following takes place: (1) The writer begins deleting characters in the text (pre-contextual or contextual deletion), or (2) the writer moves the cursor to a different location in the text and then begins producing new characters (contextual insertion).

End of the revision candidate. The end of a revision is indicated by the start of normal text production or the start of a new revision event. The start of a new revision event can be easily inferred using a rule-based algorithm. However, the point where a pre-contextual or contextual deletion is followed by normal text production not directly caused by the revision, is not trivial to automatically identify. For this, inferences need to be made based on the deleted text and the text typed after the deletion. Therefore, manual annotation is used to indicate the point where a revision ends and is followed by normal text production.

The end of the revision candidate is approximated to be at a point when the writer initiates a new revision event candidate (with the rules discussed above). Thereafter, manual annotation is used to update the end of the revision. The revision event candidate is a "placeholder" approximation of very limited utility, because it assumes that the entire process of text production is split into non-intersecting revision candidates without any fluent text production (i.e., non-revision behavior) in between. This assumption is, of course, not true. Below we discuss how the true end of revision events can be annotated manually and automatically.

Manual annotation of revision candidates

The output of the rule-based algorithm described above is a table of revision candidates that can be used for the manual annotation process. The table contains one line per revision candidate. In addition, the keystroke ID at the start of the revision candidate, the deleted characters, and the typed characters are provided to aid the manual annotation in the following step (see the first four columns in Table 1). The R code for this rule-based algorithm creating the manual annotation table can be found at https://github.com/RConijn/RevisionEvent The code has been specifically tailored to Inputlog data (Leijten & Van Waes, 2013), but can be applied to any other keystroke logging program that collects for each key pressed the key type, position in the document and current document length. A manual annotation guide is available from a previous study (Conijn et al., 2021), which includes guidelines and examples on the annotation of a true revision and revision end as well as additional revision properties which are not discussed in the present paper. The part of

Table 1 Example table of four annotated revision candidates

Revision candidate number	Keystroke ID	Removed characters	Typed characters	Revision (Y/N)	Revision end
1	0		I wriet	No	
2	7	wriet	write every day	Yes	write/ every day
3	27	every day	novels and poems. Some ti	Yes	novels and poems./ Some ti
4	61	e ti	edays I do not write	Yes	edays/ I do not write

The forward slash (/) in "Revision end" denotes the end of the revision as manually annotated

the guide detailing the annotation of a true revision and the revision end is reproduced in Appendix A.

Automated identification of revision events

As manual annotation is time-consuming, we complement the manual annotation approach with an automated approach. Here, we train a machine learning algorithm on an existing, manually annotated dataset to identify whether the revision candidate is a true revision event and to indicate where the revision ends. The machine-learning algorithms may in turn be used to automatically identify revision events in a new dataset from the rule-based approximations (without manual annotation). The application of the algorithm on a new dataset is described as a case study. The R code for running these machine-learning algorithms can be found at https://github.com/RConijn/RevisionEvent

Extracting revision events

Manual annotation of revision candidates

To refine the rule-based approximations defined in the "Rule-based approximation of revision candidates" section, four human coders manually annotated true revision events from the automatically extracted revision candidates. For each true revision event, the end of the revision was also manually annotated. For this, the revision candidate table (as shown in Table 1) was used in combination with a visual replay of the writing process and an overlaid eye fixation marker to explore the revision event in the context of the writing process. The manual coding was conducted as a part of a previous study, described in detail in (Conijn et al., 2021).

For the manual annotations, a dataset of keystroke and eye fixation data (using Gazepoint GP3 devices with 0.5–1 degree of visual angle accuracy and a 60 Hz sampling rate) was sampled from a large database yielded by the CyWrite project (Chukharev-Hudilainen, 2019; Chukharev-Hudilainen et al., 2019; Feng et al., 2016; Ranalli et al., 2018). The dataset contained texts produced by 65 students writing with CyWrite, a web-based word-processing tool. These students came from different backgrounds (graduate and undergraduate first-language writers and second-language learners) and wrote to different tasks (abstract, argumentative text). Specifically, the dataset included 20 texts from native graduate students, writing 150–250-word abstracts of a research article; 20 texts from native undergraduate students, writing an argumentative task adapted from the Test of English as a Foreign Language; and 25 texts from learners of English as a second language (most likely undergraduate students), also writing an argumentative task adapted from the Test of English as a Foreign Language. For these 65 participants, a total of 7,120 revision event candidates ($M = 110$, $SD = 53$ per participant) were indicated based on the rule-based algorithm.

For each revision candidate, one of the four annotators indicated whether this was indeed a revision (the fifth column in Table 1). For 15 randomly chosen texts, all revision candidates were annotated by two annotators (in randomized pairs) to calculate inter-rater reliability metrics. On average, 93% of the candidate revision events were indicated as a true revision (Krippendorff's $\alpha = 0.96$; percentage agreement = 99%). In addition, the annotators annotated where the revision ended, indicated with a backslash in the typed characters (see sixth column, Table 1; Krippendorff's $\alpha = 0.74$; percentage agreement = 81%). Thus, a true revision and the revision end could be manually annotated with good reliability (Krippendorff, 2004).

Automated identification of true revision events

To predict true revision events (out of revision candidates), binary classification models were trained on the keystroke variables obtained from the annotated dataset. In total, 19 predictor variables were collected that could be automatically extracted from the keystroke data. The predictor variables are related to the position and time of the start of the revision event, the duration of the revision event, the number of characters inserted and deleted, and the starting point of the revision. For an overview of all the predictor variables, see Appendix B.

Three different machine learning models were trained using the 'caret' R-package (Kuhn, 2019). Specifically, random forests and support vector machines with a radial kernel were chosen as they generally work well on continuous data. In addition, a decision tree model was used to determine whether a simpler, more intuitive model would perform similarly well. Majority class prediction (93% is annotated as a true revision event) was used as a baseline. All predictor variables were centered and scaled as a pre-processing step. Parameter tuning was done via tenfold cross-validation, optimizing for the largest F_1-score. Accuracy, precision, recall, and F_1-score on the ten folds are presented as evaluation metrics.

The outcomes of the machine learning algorithms are presented in Table 2. All models outperform the majority class baseline, indicating that the models work better than just predicting every revision candidate as a true revision event. The decision tree and the random forest proved to be the best algorithms, with similar performances (F_1-score = 0.966 and 0.963, respectively). Both the decision tree and random forest models show high precision, indicating that almost all predictions of true revision events were indeed a revision event (only a few false positives). The

Table 2 Performance of the prediction of a true revision event

	Accuracy (%)	Precision	Recall	F_1-score
Support vector machine	97.1	.774 (.059)	.771 (.061)	.771 (.047)
Random forest	99.5	.988 (.020)	**.940** (.036)	.963 (.024)
Decision tree	**99.6**	**.998** (.007)	.937 (.036)	**.966** (.018)
Majority class (baseline)	93.3	.933	1.000	.965

Means across the 10 folds are reported, with standard deviations in parentheses. Bold indicates the highest score

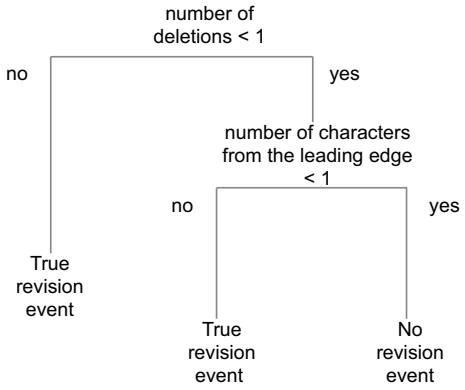

Fig. 1 Decision tree for the prediction of a true revision event

recall was slightly lower, indicating that some revisions were still missed (false negatives). As the decision tree algorithm is also the easiest to interpret, we opted for the decision tree as our final model.

We further inspected this algorithm to get more insights into the workings of the model. First, the confusion matrix showed that although the model worked fairly well, the model resulted in slightly more false positives (non-revisions indicated as revision; 0.40%), compared to false negatives (0.01%). Second, the plot of the decision tree (see Fig. 1) showed that the model was relatively simple, based on only two rules: if the number of deletions was lower than 1 (i.e., there were no deletions during the candidate revision event) and the number of characters from the leading edge was lower than 1, it was *not* a true revision event, otherwise, it was a revision event. That is, if the writer started inserting text at the leading edge, it is not considered as a revision event. Combined with our rule-based approximation, we can now update the definition of a revision event as follows: A revision event starts if one of the following takes place: (1) The writer begins deleting characters in the text, or (2) the writer moves the cursor to a different location in the text, *except for the leading edge (excluding invisible characters)*, and then begins producing new characters. Hence, the updated rule excludes all insertions that are added at the end of the text, as these are seen as new text production. Thus, for the automatic extraction of true revision events, we do not need a complex machine learning algorithm, but can reach sufficient performance with a rule-based algorithm.

Automated identification of revision end

For the end of the revision, machine learning algorithms were trained on keystroke and eye-tracking variables obtained from the annotated dataset. Here, we only focused on the revision candidates that were annotated as true revision events (6,641 revisions). To predict the location of the revision end (i.e., the point where the writer resumed fluent outputting or initiated a new revision), there are two potential approaches. The first approach is predicting the length of the revision (regression); e.g., in Table 1, revision number 2, the revision length is five characters. The second approach is predicting for each keystroke whether

the revision ended there (binary classification). As binary classification generally results in higher accuracy compared to regression, we opted for the second approach. For this, the annotated table was collapsed into a dataset showing a keystroke per row. For each keystroke, the algorithm indicated the probability of the specific keystroke being the final keystroke of the revision, that is, the revision end. Here, any probability equal to or higher than 50% might be considered a revision end. As there is only one revision end per revision number, the keystroke with the highest probability of being a revision end is selected for each revision number.

For the prediction of revision end, several machine learning algorithms were trained on keystroke data only and keystroke data in combination with eye-tracking data. In this way, we could determine the added value of using eye-tracking for the prediction of revision end. In total, 37 predictor variables were collected that could be automatically extracted from the keystroke data and the eye-tracking data (see Appendix B). The same 19 predictor variables from prediction start were included (Table 2). In addition, 12 keystroke and 6 eye-tracking variables related to a specific keystroke were added. These consisted of information related to pauses between keystrokes, the string distance (restricted Damerau-Levenshtein distance; Van der Loo, 2014) between the deleted and the typed characters (so far), eye fixations, and saccades.

Similarly to the prediction of a true revision event, three machine learning models (random forests, support vector machines with radial kernel, and decision trees) were trained on the datasets with only keystroke features, and with keystroke and eye-tracking features combined. As baseline, we used the simple rule where the revision end was the keystroke where the number of typed characters (so far) was equal to the number of deleted characters (or the final keystroke if the number of typed characters was always lower than the number of deleted characters). The dataset was split into 70% of the files for training, and 30% of the files for testing. In other words, none of the revisions from a single user were available in both the training and the testing set. All predictor variables were standardized as a pre-processing step. As there were many missing variables for the eye-tracking data (i.e., not every keystroke was preceded by a fixation or saccade), an additional variable was added to indicate that there was no eye data present for that specific keystroke and all missing values were set to zero. Parameter tuning was done via tenfold cross-validation, optimizing for the largest F_1-score. Then, the best model was run on the test set, and the keystroke with the highest confidence score was selected as revision end. Accuracy (exact and one off), precision, recall, and F_1-score on the test set are presented as evaluation metrics.

The outcomes of the machine learning algorithms are presented in Table 3. All models, except the decision tree models, outperformed the baseline. The support vector machines slightly outperformed the baseline with 43/47% of the revision end predicted at the exact location of the revision end, and 60/65% within one keystroke from the revision end, for keystroke and eye-tracking data or keystroke data only, respectively. The revision was on average 3.5–3.6 ($SD = 9.0$) characters away from the actual revision end, indicating that the support vector machine models, even though they outperformed the baseline, were not very accurate. The random

Table 3 Performance of the prediction of revision end

	Feature set	Accuracy exact	Accuracy one off	Accuracy (chars away M, SD)	Precision	Recall	F_1-score
Support vector machine	Key	.473	.645	3.5 (9.0)	.975	.974	.975
Random forest	Key	.713	**.830**	1.8 (6.6)	.987	**.986**	.986
Decision tree	Key	.240	.420	4.1 (6.9)	**.991**	.837	.908
Support vector machine	Key + eye	.432	.601	3.6 (9.0)	.973	.972	.972
Random forest	Key + eye	**.714**	.829	**1.8 (6.4)**	.987	**.986**	**.987**
Decision tree	Key + eye	.240	.420	4.1 (6.9)	**.991**	.837	.908
Typed = deleted (baseline)	-	.360	.570	2.9 (6.6)	.974	.937	.955

Scores are reported for the test set. Bold indicates the highest score

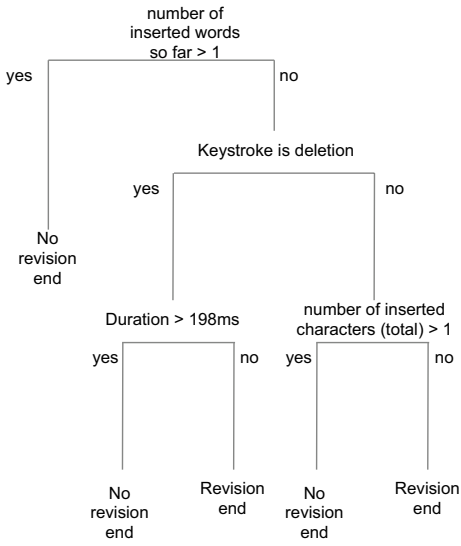

Fig. 2 Decision tree for the prediction of revision end (note that accuracy was rather low)

forest models proved to be the best performing models, with 71% of the revision end predicted at the exact location of the revision end, and 83% of the revision end predicted within one keystroke from the revision end. On average, the revision end was predicted 1.8 characters away from the actual revision end, with a quite large standard deviation of 6.4–6.6 (depending on the feature set). This indicated that even though the prediction is quite accurate, at some points the prediction is still rather far away from the actual revision end.

Interestingly, even though the baseline can be considered a simple decision tree, the decision tree models did not outperform the baseline. In fact, the decision tree actually performed less well than the baseline. This discrepancy might be explained by the setup. In our setup, the decision tree predicts for every keystroke whether it is the revision end, where the keystroke with the highest probability will be chosen as the actual revision end. In contrast, the baseline only considers a single keystroke as revision end (i.e., the keystroke where the number of typed characters are equal to the number of deleted characters). In addition, the decision tree model does not directly compare the input features, while the baseline model does. Within the decision tree (Fig. 2), the number of typed and deleted characters are important, as they do show up separately in the model. However, their values are not compared. At the root of the tree, it is shown that if the number of inserted words is larger than 1, there is no revision end. This indicates that the tree always predicts the revision end to be within or after the first word typed, resulting in only revisions of at most one word. This might be one reason why the decision tree performs poorly. Furthermore, within the first word, the revision end is predicted immediately after a deletion if the duration of the revision event is very short (<198 ms), otherwise the revision end is predicted if the number of inserted characters is equal to 1. Combined, this shows that this decision tree model is too simple to accurately predict the revision end.

For almost all models, the performance was similar for the feature set with only keystroke data, compared to the feature set with both keystroke and eye-tracking data. This indicates that the eye-tracking data had little additional value on top of the keystroke data in predicting the revision end. This might indicate that the eyetracking with Gazepoint GP3 devices was not accurate enough for the current purpose, or might indicate that eye tracking has limited added value in predicting revision events. Hence, logging eye tracking data may not necessarily be needed for applying this model in future work. For the support vector machines, the model with only keystroke data even outperformed the model with both keystroke and eye-tracking data. This might indicate that the model with both keystroke and eye tracking data was too complex, consisting of too many features, potentially resulting in overfitting.

To gain more insight into the errors made by the best performing model (random forest), we took a closer look at the cases where the prediction was very far away from the revision end. Here, it was shown that the largest errors were made when the model predicted a rather small revision (e.g., single character), while the manual annotation showed a large revision (e.g., revision of a full sentence) or vice versa. An example of three revisions made by participant 43 is shown in Table 4. Here, at revision number 51, the revision end is manually indicated after "I", arguably because the writer forgot to capitalize the "i". The algorithm predicts the revision end only at the end of that sentence, indicating that the revision might have been larger. This might be true if, for example, the writer had planned to write ("Two summers ago in Ireland"), later on revised to ("Two summers ago I took …"); however, the revision to capitalize the "i" might be more plausible here. For the next revision (revision number 52), it is somewhat more complex. Here, the manual annotator indicates the revision ends after the writer has replaced "W" at the start of the sentence by "The". Here the algorithm again predicts that the revision ends after the end of the sentence, indicating that the revision is a sentence revision rather than a word revision. This example already shows that the errors might sometimes also be explained by cases which were manually also hard to define.

To gain more insight into the models and the added value of the eye-tracking data, feature importance was calculated for the random forest model with the keystroke and eye-tracking data (see Fig. 3). Two of the most important features for the model were features related to the full revision event (duration and number of inserted characters), as indicated by the rule-based approximation. Thereafter, several features were important related to specific keystrokes within the revision, such as the number of inserted characters and words so far, interkeystroke interval, and the string distance between the typed and deleted characters. The features related to eye-tracking showed limited importance, with fixation duration and saccade length being only the twelfth and fourteenth most important predictors, respectively. This corroborates with previous findings that the addition of eye tracking data showed limited improvement in the models, on top of the keystroke data. If we compare the features with the prediction of revision end, we can see that none of the predictors for a true revision (number of deleted characters and number of characters from the leading edge) are important for the prediction of revision end. Hence, a true revision event and a revision end show distinctive properties.

Table 4 Example of (large) errors made for the prediction of revision end within the random forest model with keystroke and eye tracking data (\ indicates revision end)

Rev. no	Removed characters	Typed characters	Revision end (manual)	Revision end (algorithm)	Difference in characters
		Two summers ago i			
51	i	I took a ten day trip to Ireland. W	I\ took a ten day trip to Ireland. W	I took a ten day trip to Ireland.\ W	−32
52	W	The day before I left one of my favorite bands had come out with a new album. I listened to it non stop the eentire	The\ day before I left one of my favorite bands had come out with a new album. I listened to it non stop the eentire	The day before I left one of my favorite bands had come out with a new album.\ I listened to it non stop the eentire	−74
53	entire	ntire trip	ntire\ trip	ntire\ trip	0
		…			

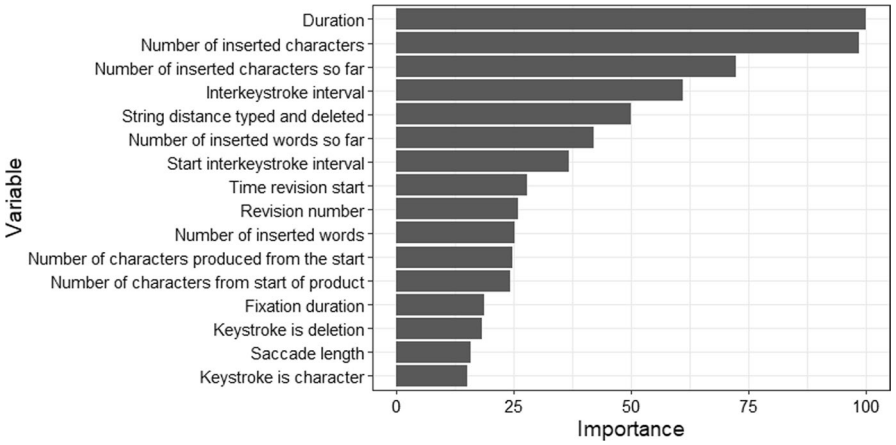

Fig. 3 Feature importance of the prediction of the revision end with the Random Forest algorithm and the keystroke and eye data. *Note*. Importance indicates relative importance to the most important feature; only the 16 most important features are shown

Proof of concept case study

As a proof of concept, we ran the best performing models for the prediction of a true revision event and the revision end on a new dataset. Here, we aim to show how the algorithms can be applied to a new (unseen) dataset and how this could be used to compare revision events across groups. Here, we compare the revision events for students writing in their first language (L1) versus their second language (L2). The dataset was collected for a previous study (Chukharev-Hudilainen et al., 2019) and consisted of keystroke and eye tracking data from 24 undergraduate students at a private university in Turkey writing two essays. One of the essays was written in Turkish (L1), and the other was written in English (L2), with prompt-to-language assignment and task order counterbalanced. The prompts for both essays were adapted from the Test of English as a Foreign Language (TOEFL). Participants were allowed up to 40 min to write each essay, but they could finish writing earlier if they were satisfied with their essay. There was no minimum or maximum length requirement for their texts.

First, the rule-based algorithm was run to determine the revision candidates. In total, 5326 revision candidates were identified within the 48 essays. Thereafter, the decision tree model was used (with the updated revision event rules), to determine the true revision events. This resulted in a total of 4,936 (93%) true revision events. A similar percentage was found for the manually annotated dataset, where also 93% of the revision candidates were manually annotated as a revision ("Manual annotation of revision candidates" section). Second, for all true revision events, the revision end was indicated, using the best performing algorithm from the "Automated identification of true revision events" section: the random forest model containing both keystroke and eye tracking data.

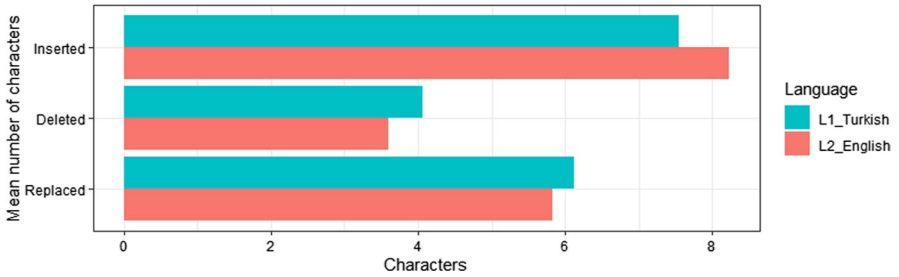

Fig. 4 Average revision length for insertions (inserted) and deletion revisions (deleted and replaced) in L1 and L2

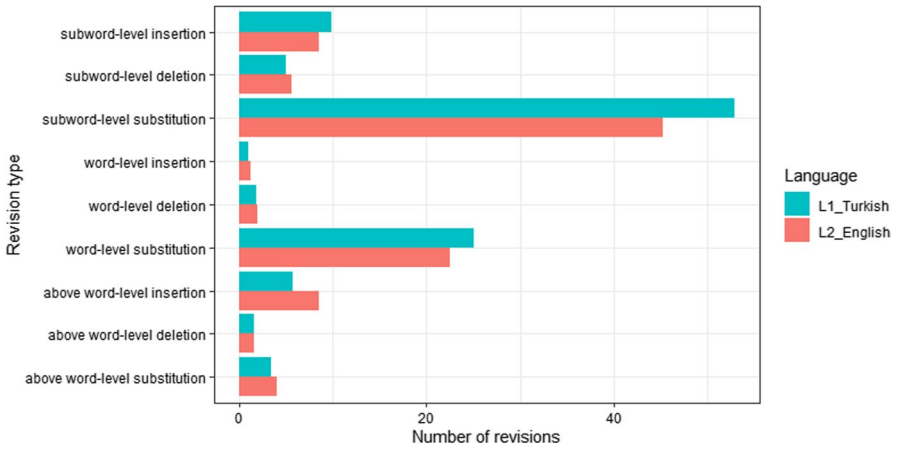

Fig. 5 Number of revisions per revision type for L1 and L2

The automatically extracted revision events allow us to compare the revisions across the two languages in our dataset (L1 and L2). The results showed that the participants made slightly more revisions in L1, Turkish, with a large variance in the number of revisions ($M = 106$, $SD = 73$), compared to L2, English ($M = 99$, $SD = 40$). The lower number of revisions in L2, might be related to the fact that the participants also wrote less in L2 ($M = 1458$, $SD = 518$ characters) compared to L1 ($M = 1871$, $SD = 847$ characters).

The length of the revision events (including the inserted characters for insertions and the deleted and replaced characters for deletions) was similar for students writing in L1 ($M = 9.7$, $SD = 12.4$) and L2 ($M = 9.1$, $SD = 10.3$). In both L1 and L2, students' insertions were on average seven to eight characters long, while their deletions consisted of three to four characters, which were replaced by a mean of six characters (see Fig. 4).

With a clear indication of the revision start and revision end, we could use additional rules to identify how specific properties of these revisions, such as the scope of the revisions, differed between L1 and L2. Specifically, for each revision event, it was indicated whether the scope of the revision was below word-level (at

most three characters within a single word), at word-level (at most two words), and above word-level (more than two words). In addition, the action of the revision was indicated: an insertion, a deletion, or a substitution. The resulst showed only some small differences in the type of revisions made (see Fig. 5). Subword-level revisions were the most frequent revisions in both L1 and L2, with slightly fewer subword-level substitutions in L2, compared to L1. In addition, the participants made slightly more above word-level insertions in L2 compared to L1. To conclude, this case study shows that the automated revision event extraction could be used to provide more (and automated) insights into the revisions made in a new dataset.

Discussion

In the current paper we provided a method to extract the *full* process of each revision event within a keystroke log. This full process is based on two characteristics: the revision start and revision end. In contrast to previous approaches (e.g., only focusing on deletions; Van Waes et al., 2014), we adopted a rather broad definition of revisions, which included revisions started by pre-contextual and contextual deletions, as well as contextual insertions. In addition, the revision does not end at the last delete keystroke (*cf.* events, Baaijen et al., 2012; insertion and revision bursts, Baaijen & Galbraith, 2018; or S-notation, Kollberg, 1996). Rather, the revision ends when the cognitive process of making a revision has been finished. The approach presented consists both of a rule-based approximation which can be used for manual annotation, as well as an automated extraction. The rule-based approximation with manual coding is generally more accurate compared to the automated approach, but still needs relatively time-intensive manual annotation (although the rule-based approximation already simplifies the process to a large extent). In contrast, the automated approach is much faster, but requires some knowledge of scripting (e.g., in R) and machine learning. By providing both a manual and an automated approach, with accompanying scripts, we envision this approach can be easily adopted by other researchers.

The automated approach showed that the true revision event could be automatically detected via a decision tree algorithm with high accuracy. Hence, for the prediction of a true revision event, no complex machine learning models are needed, but a simple rule-based algorithm already shows high accuracy. The prediction of the revision end showed to be slightly more complex. The random forest model proved to result in the highest accuracy, with 71% of the revision end predicted at the exact location of the revision end. The predicted revision end was on average 1.8 characters away from the actual revision end, with a standard deviation of 6.4 characters. This indicates that several predictions were still very far away from the actual revision end. Further inspections of the errors showed that these often consisted of cases where a character or word revision was made, while the algorithm predicted the revision of a full sentence (or vice versa). This can be explained by the fact that it is sometimes difficult to annotate revisions in these cases. If only the first character of a word is typed, it is hard to infer what the writer intended to write before they started the revision. This can also be seen in the lower inter-rater reliability for the prediction of revision

end (Krippendorff's $\alpha = 0.74$). Data triangulation (e.g., concurrent think-aloud or retrospective stimulated recall) might result in a better understanding of the intended words and the full length of the revision, making for a more reliable manual annotated dataset to train the machine learning algorithms. This could, in turn, also provide more insight into *why* the revision occurred (Wengelin et al., 2019).

The case study showed that the current models can be applied to a new dataset, allowing for automated in-depth analyses of revision events, such as the level or scope of the revision. In future work, these levels of revisions could be combined with the spatial location of the revision in the text as well as the temporal location of the revision in the writing process, to provide a full overview of the where and when certain types of revisions are being made.

Limitations

The current study is limited in two ways. First, for the prediction of a true revision, we aimed to use the manual annotation to improve upon the operationalization of the leading edge (the last character or the text produced so far). Here, we argued this operationalization would fail if there are invisible characters at the end of the text (Lindgren et al., 2019), or when there is trailing text which is not used in the analyzed writing session (e.g., a bibliography section). Hence, the updated rule focused on any change at the point of inscription, except from text insertions at the leading edge (excluding the invisible characters). Especially in more complex revision operations, including and reordering and restructuring, this approach results in many (smaller) revision events, as opposed to a single reordering revision. In addition, when a larger revision is interrupted by a smaller revision, the revision will be split into two revision events. Future work should determine how the timing and the spatial location of the revision event can be used to identify sequences of related revision events (or revision episodes; Kollberg, 1996).

A second and related limitation is that the manually annotated dataset that was used to train the prediction algorithms, consisted of relatively short texts, collected within short writing sessions. Hence, most of the revisions were also relatively small in size (i.e., many spelling and grammar revisions). Accordingly, the current model for predicting the revision end might be too much tailored to short revisions. The prediction models might perform more poorly on new datasets which originate from longer writing sessions with extensive reordering and restructuring. Hence, future work should test the model on additional datasets that are more diverse in the types of revision to determine how well the models generalize to other types of text. Despite the fact that the models might generalize less well toward other types of texts, we do feel the current approach is valuable in two ways. Currently, most texts analyzed in writing research consist of short, single session writing tasks. Therefore, the current automated approach will be useful in the majority of writing studies. In addition, for the longer, multi-session writing tasks, the rule-based approximation in combination with the manual annotation still provides a useful alternative.

Implications for writing research

In the current paper, we showed several ways in which a data-driven approach can aid theory-driven writing research. First, we showed that machine learning can be used to automatically extract full revisions from keystroke data without time-intensive manual annotations needed. Although the current system only works for Inputlog and CyWrite data, it can be easily adapted to other keystroke logging and eye tracking applications. In addition, the automatic annotation could also allow for the integration of revision identification in real-time systems that automatically provide information on the writer's evolving writing process, such as writing dashboards or automated writing feedback systems (see e.g., Conijn et al., 2020; Ranalli et al., 2018).

Second, a data-driven approach can provide more insight into theoretical constructs in writing. Specifically, we showed how the algorithms could provide more insight into our definition of revision, or the underlying 'rules' used in manual annotations. For example, for the prediction of a revision event, we showed that our initial definition could be improved by adding a single rule. In this way, data-driven approaches could be used to shed light on constructs that are theoretically hard to define.

Third, with the case study we showed that by considering the full revision event, rather than only the deletion, more information can be obtained about the nature of the revision. This makes it easier to automatically classify different types of revisions (see e.g., Conijn et al., 2019; Daxenberger & Gurevych, 2013), or to identify sequences of revisions (see e.g., Kollberg, 1996). This could be used to either focus on certain types of revisions, or to exclude certain types of revisions, which would be of less interest or could confound the analysis. For example, by using the spatial location of the revision event in combination with the number of characters deleted or inserted, researchers might distinguish between editing and reviewing revision events (*cf.* Flower & Hayes, 1980).

Lastly, the table of revision events can be used to provide a more condensed overview of the rather 'detailed' keystroke log, which can be used for a more insightful unit of analysis (as opposed to a single keystroke) within sequential analyses to gain insight into patterns of revisions (*cf.* Zhang et al., 2016).

Conclusion

The current paper provided a method to extract the full process of each revision event within a keystroke log using both manual annotation and machine learning algorithms. This showed how computational approaches can be beneficial in providing more detailed measurements of revision in writing. Specifically, the case study showed that this approach can be used to automatically gain more insight into the nature of revisions. In addition, inspection of the models resulted in additional understanding of the underlying rules in our manual annotation. To conclude, this illustrates how data-driven methods can aid theory-driven research.

Appendix A: Manual annotation of revisions (adapted from Conijn et al., 2021)

Table 5 Annotation example for revision start [Y/N]

RID	Removed characters	Typed characters	True revision	Revision end
1		to	0	
2	to	To hear myself	1	To/ hear myself

Table 6 Annotation example for revision end

RID	Removed characters	Typed characters	True revision	Revision end
1	I wriet	I write every day	1	I write/ every day
2	Every day	Novels and poems/	1	Novels and poems!./

True revision [Y/N] The revision candidate is a true revision, unless it is just fluent text production at the leading edge (cursor location). Values: 0,1. For an example see Table 5.

For every revision candidate that is coded as a true revision you need to indicate the revision end.

Revision end All characters typed up to where the revision ended and text production started. If the writer only revised a character within a word, the revision ends at the end of that word. If the writer revised a word/phrase/sentence to follow a new train of thought, be lenient: put the revision end as far as possible. If there is already a slash in revision end (because the writer typed a slash), please change the typed slash into an exclamation mark (!). Values: put a slash where the revision ended (see Table 6).

Appendix B: Descriptive statistics of all predictor variables

See Tables 7, 8

Table 7 Descriptive statistics for the variables used to predict a true revision event (N = 7,120)

Type	Variable	Mean	SD
Key	Revision number	70.0	51.1
	Number of characters from the leading edge (excl. invisible characters)	67.8	256.5
	Number of words from the leading edge (excl. invisible characters)	11.7	44.9
	Number of sentences from the leading edge (excl. invisible characters)	0.5	2.1
	Number of characters from the start of the writing product	893.6	652.6
	Number of characters produced up to start of the revision	891.2	652.9
	Interkeystroke interval at start of revision (ms)	2325.8	7208.0
	Time revision start (s)	579.0	458.2
	Duration (ms)	3058.8	6394.4
	Number of backspace keys	2.2	2.1
	Number of deleted characters	3.3	7.3
	Number of inserted characters	18.6	23.7
	Number of deleted words	1.0	1.3
	Number of inserted words	3.7	4.2
	Number of deleted sentences	1.0	0.1
	No change (same characters were deleted as were inserted)	7%	26%
	Revision starts without movement of cursor	81%	39%
	Starts with capital	10%	31%
	Starts with space	1%	10%
	Starts with period/comma	6%	24%

Table 8 Descriptive statistics for the keystroke-specific variables used to predict revision end (N = 143,838 keystrokes)

Type	Variable	Mean	SD
Key	Number of deletion actions (total)	4.4	7.7
	Interkeystroke interval (ms)	443.1	1961.1
	Typed character is alphanumeric character	69%	46%
	Typed character is capital	1%	11%
	Typed character is space	14%	35%
	Typed character is period or comma	1%	12%
	Keystroke is deletion	14%	35%
	Number of typed characters (so far) isequal to deleted characters	4%	19%
	Typed text (so far) is equal to deleted text	2%	12%
	String distance between typed and deleted characters	22.1	27.5
	Improvement of string distance (compared to previous keystroke)	0.7	0.6
	Number of characters inserted so far	21.5	28.4
	Number of words inserted so far	4.2	4.9
Eye	No eye data present	78%	42%
	Fixation duration (ms)	478.4	350.7
	Saccade length (number of characters)	2.3	99.6
	Saccade is initiated at leading edge	27%	44%
	Static eye movement (saccade length = 0)	7%	26%
	Progressive eye movement (saccade length > 0)	60%	49%
	Regressive eye movement (saccade length < 0)	8%	47%

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. 2016868. In addition, we would like to thank Bauke Conijn, Jens Roeser, and Luuk Van Waes for their assistance in data cleaning and modeling and making the analysis compatible with Inputlog.

References

Baaijen, V. M., & Galbraith, D. (2018). Discovery through writing: relationships with writing processes and text quality. *Cognition and Instruction, 36*(3), 1–25. https://doi.org/10.1080/07370008.2018.1456431

Baaijen, V. M., Galbraith, D., & de Glopper, K. (2012). Keystroke analysis: reflections on procedures and measures. *Written Communication, 29*(3), 246–277. https://doi.org/10.1177/0741088312451108

Chukharev-Hudilainen, E. (2019). Empowering automated writing evaluation with keystroke logging. In E. Lindgren & K. Sullivan (Eds.), *Observing Writing* (Vol. 38, pp. 125–142). Brill.

Chukharev-Hudilainen, E., Saricaoglu, A., Torrance, M., & Feng, H. H. (2019). Combined deployable keystroke logging and eyetracking for investigating L2 writing fluency. *Studies in Second Language Acquisition*, *41*(3), 583–604.

Conijn, R., Speltz, E. D., van Zaanen, M., Waes, L. V., & Chukharev-Hudilainen, E. (2021). A product- and process-oriented tagset for revisions in writing. *Written Communication*. https://doi.org/10.1177/07410883211052104.

Conijn, R., Van Waes, L., & van Zaanen, M. (2020). Human-centered design of a dashboard on students' revisions during writing. *Conference Proceedings of the 14th European Conference on Technology Enhanced Learning, EC-TEL*. https://doi.org/10.1007/978-3-030-57717-9_3

Conijn, R., van Zaanen, M., Leijten, M., & Van Waes, L. (2019). How to typo? Building a process-based model of typographic error revisions. *Journal of Writing Analytics, 3*, 1.

Daxenberger, J., & Gurevych, I. (2013). Automatically classifying edit categories in Wikipedia revisions. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 578–589.

Eklundh, K. S., & Kollberg, P. (2003). Emerging discourse structure: Computer-assisted episode analysis as a window to global revision in university students' writing. *Journal of Pragmatics, 35*(6), 869–891. https://doi.org/10.1016/S0378-2166(02)00123-6

Feng, H. H., Saricaoglu, A., & Chukharev-Hudilainen, E. (2016). Automated error detection for developing grammar proficiency of esl learners. *Calico Journal*, *33*(1), 49–70. https://www.jstor.org/stable/calicojournal.33.1.49

Fitzgerald, J. (1987). Research on revision in writing. *Review of Educational Research, 57*(4), 481–506. https://doi.org/10.3102/00346543057004481

Flower, L., & Hayes, J. R. (1980). The cognition of discovery: Defining a rhetorical problem. *College Composition and Communication, 31*(1), 21–32.

Galbraith, D., & Baaijen, V. M. (2019). Aligning keystrokes with cognitive processes in writing. In E. Lindgren & K. Sullivan (Eds.), *Observing Writing* (Vol. 38, pp. 306–325). Brill. https://doi.org/10.1163/9789004392526_015

Horning, A., & Becker, A. (2006). *Revision: History, theory, and practice*. Parlor Press LLC.

Kaufer, D. S., Hayes, J. R., & Flower, L. (1986). Composing written sentences. *Research in the Teaching of English, 1*, 121–140.

Kollberg, P. (1996). S-notation as a tool for analysing the episodic structure of revisions. *European Writing Conferences*, 1–15.

Krippendorff, K. (2004). *Content analysis: An introduction to its methodology.* (Second Edition). Sage.

Kuhn, M. (2019). *caret: Classification and Regression Training (R package version 6.0–84)*. https://CRAN.R-project.org/package=caret

Leijten, M., & Van Waes, L. (2013). Keystroke logging in writing research: Using Inputlog to analyze and visualize writing processes. *Written Communication, 30*(3), 358–392. https://doi.org/10.1177/0741088313491692

Leijten, M., Van Waes, L., & Van Horenbeeck, E. (2019). *Inputlog Manual*. https://www.inputlog.net/wp-content/uploads/Inputlog_manual.pdf

Lindgren, E., & Sullivan, K. P. (2006a). Analysing online revision. In K. P. Sullivan & E. Lindgren (Eds.), *Computer keystroke logging and writing: methods and applications (studies in writing)* (pp. 157–188). Elsevier.

Lindgren, E., & Sullivan, K. P. (2006b). Writing and the analysis of revision: An overview. In K. P. Sullivan & E. Lindgren (Eds.), *Computer keystroke logging and writing: Methods and applications (Studies in Writing)* (pp. 31–40). Elsevier.

Lindgren, E., & Sullivan, K. P. (2019). Observing Writing: Insights from Keystroke Logging and Handwriting. *Brill*. https://doi.org/10.1163/9789004392526

Lindgren, E., Westum, A., Outakoski, H., & Sullivan, K. P. H. (2019). Revising at the Leading Edge: Shaping Ideas or Clearing up Noise. In E. Lindgren & K. Sullivan (Eds.), *Observing Writing* (Vol. 38, pp. 346–365). Brill. https://doi.org/10.1163/9789004392526_017

Medimorec, S., & Risko, E. F. (2017). Pauses in written composition: On the importance of where writers pause. *Reading and Writing, 30*(6), 1267–1285. https://doi.org/10.1007/s11145-017-9723-7

Murray, D. M. (1978). Internal revision: A process of discovery. *Research on Composing: Points of Departure, 1*, 85–103.

Olive, T. (2014). Toward a parallel and cascading model of the writing system: a review of research on writing processes coordination. *Journal of Writing Research, 6*, 173–194.

Ranalli, J., Feng, H.-H., & Chukharev-Hudilainen, E. (2018). Exploring the potential of process-tracing technologies to support assessment for learning of L2 writing. *Assessing Writing, 36*, 77–89. https://doi.org/10.1016/j.asw.2018.03.007

Roeser, J., Torrance, M., & Baguley, T. (2019). Advance planning in written and spoken sentence production. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 45*(11), 1983. https://doi.org/10.1037/xlm0000685

Severinson–Eklundh, K., & Kollberg, P. (2001). Studying writers' revision patterns with S-notation analysis. In T. Olive & C. M. Levy (Eds.), *Contemporary Tools and Techniques for Studying Writing* (Vol. 10, pp. 89–104). Springer.

Van der Loo, M. P. (2014). The stringdist package for approximate string matching. *The R Journal, 6*(1), 111–122.

Van Waes, L., Leijten, M., Wengelin, Å., & Lindgren, E. (2012). Logging tools to study digital writing processes. In V. W. Berninger (Ed.), *Past, present, and future contributions of cognitive writing research to cognitive psychology* (pp. 507–533). Psychology Press.

Van Waes, L., van Weijen, D., & Leijten, M. (2014). Learning to write in an online writing center: The effect of learning styles on the writing process. *Computers & Education, 73*, 60–71. https://doi.org/10.1016/j.compedu.2013.12.009

Wengelin, Å., Frid, J., Johansson, R., & Johansson, V. (2019). Combining keystroke logging with other methods: Towards an experimental environment for writing process research. In *Observing Writing* (pp. 30–49). Brill. https://doi.org/10.1163/9789004392526_003

Zhang, M., Hao, J., Li, C., & Deane, P. (2016). Classification of Writing Patterns Using Keystroke Logs. In L. A. van der Ark, D. M. Bolt, W.-C. Wang, J. A. Douglas, & M. Wiberg (Eds.), *Quantitative Psychology Research: The 80th Annual Meeting of the Psychometric Society, Beijing, 2015* (pp. 299–314). Springer. https://doi.org/10.1007/978-3-319-38759-8_23