languages

MDPI

Article

Automated Extraction and Analysis of Sentences under Production: A Theoretical Framework and Its Evaluation

Malgorzata Anna Ulasik [1,2,*] and Aleksandra Miletić [3]

[1] ZHAW School of Applied Linguistics, Institute of Language Competence, Theaterstrasse 17, 8400 Winterthur, Switzerland
[2] Faculty of Arts, Department of Language and Information Sciences, University of Lausanne, Bâtiment Anthropole, 1015 Lausanne, Switzerland
[3] Department of Digital Humanities, University of Helsinki, P.O. Box 4, Yliopistonkatu 4, 00100 Helsinki, Finland; aleksandramiletic1207@gmail.com
* Correspondence: malgorzataanna.ulasik@zhaw.ch

Abstract: Sentences are generally understood to be essential communicative units in writing that are built to express thoughts and meanings. Studying sentence production provides a valuable opportunity to shed new light on the writing process itself and on the underlying cognitive processes. Nevertheless, research on the production of sentences in writing remains scarce. We propose a theoretical framework and an open-source implementation that aim to facilitate the study of sentence production based on keystroke logs. We centre our approach around the notion of sentence history: all the versions of a given sentence during the production of a text. The implementation takes keystroke logs as input and extracts sentence versions, aggregates them into sentence histories and evaluates the sentencehood of each sentence version. We provide detailed evaluation of the implementation based on a manually annotated corpus of texts in French, German and English. The implementation yields strong results on the three processing aspects.

Keywords: writing process; keystroke logging; sentence production; text history; sentence history; linguistic modelling

check for updates

Citation: Ulasik, Malgorzata Anna, and Aleksandra Miletić. 2024. Automated Extraction and Analysis of Sentences under Production: A Theoretical Framework and Its Evaluation. *Languages* 9: 71. https://doi.org/10.3390/languages9030071

Academic Editors: Georgeta Cislaru and Philippe Martin

Received: 3 October 2023
Revised: 31 January 2024
Accepted: 1 February 2024
Published: 22 February 2024

1. Introduction

It is generally understood that a written text is composed of sentences. Sentences are considered to be essential communicative units built to express thoughts and meanings (Bühler 1918). A number of authors have analysed their nature and function (Gardiner 1922; Noreen 1903; Wundt 1922; Ries; Ries 1927, 1931; Allerton 1969; Alston 1964; Bloomfield 1933; Cinato 2018; Matthews 1993; Panther and Köpcke 2008; Paul 2010; Sauerland 2016, inter alia). Nevertheless, research on the production of sentences in writing remains scarce. Studying sentence production is particularly relevant from two points of view. First, combining behavioural data typically used in writing process research with linguistic information on the sentence level could shed new light on the writing process itself. It opens an opportunity to investigate the writing strategies in more detail by analysing the sentence production flow or examining revisions on word and sentence level. It could also allow for an investigation of writing difficulties in relation to the linguistic structures (e.g., their complexity). Second, examining text production on this level also provides an opportunity to track the process of transforming thoughts into linguistic output[1] and to observe how these thoughts evolve during writing. By observing the writer's decisions on the word and sentence levels (such as replacing words, changing syntactic structures, shortening sentences, or merging them), we can collect insights into how the initial idea which triggered a sentence construction continues evolving during writing. This makes the analysis of written sentences under production a highly interesting research area.

One widely used method for recording the writing process is keystroke logging. It stores each action performed through a keyboard as an isolated event. The action produced by the writer (adding or deleting a character) is typically accompanied by the position of the event in the text (its offset) and behavioural information on the writing process (start and end times of the event). The output of keystroke logging is sequential since the events are recorded in chronological order. However, it is not linear: writers tend to move back and forth through the text during the writing process, and two chronologically sequential events can take place at non-adjacent positions in the text.

An illustration of this fact can be observed in Table 1. In events 1–24, the writer produces a continuous sequence of characters *mieux connue sous le nom* ('better known under the name') at positions 445–469 in the document. The writer then goes back to position 449 and performs a series of backward deletions removing characters at positions 449–445 and thus deleting the word *mieux* ('better') (events 25–30). They then insert the word *aussi* ('also') at positions 445–446 (events 32–35) and finally resume writing at the end of the text by inserting *de* ('of') at positions 470–472 (events 36–38). At the end of this sequence of events, the produced text reads *aussi connue sous le nom de* ('also known under the name of').

This brief extract from a keystroke log illustrates why the string of events cannot be straightforwardly split into linguistic units. Rather, identifying linguistic units on any level (words, word phrases, sentences, paragraphs) requires reconstructing the text under production by calculating each event's position in the text and the impact it has on the text produced so far. For example, the deletion in event 26 removing the character at position 449 shifts all the characters in positions 450 and onwards one place to the left so that after this event, the character initially produced at position 450 actually occupies position 449, the one from position 449 moves to 448, and so on. The subsequent deletions produce the same effect, and the insertions recorded in events 31–35 similarly shift the subsequent characters one place to the right. In other words, the events which do not occur at the end of the text under production (illustrated in our example by events 26–35) affect the position of all the subsequent characters. Each of these changes needs to be taken into account in order to reconstruct the text at a given point of the production process.

This example also illustrates an intuitive understanding about the writing process: a text passes through a number of different versions during its genesis. In the case above, the phrase *mieux connue sous le nom* ('better known under the name') was transformed into *connue sous le nom* ('known under the name'), and then into *aussi connue sous le nom* ('also known under the name'). The collection of these intermediate versions reflects the evolution of a text. However, the intermediate versions often contain linguistically incomplete and/or ill-formed linguistic content, which makes automated processing and analysis challenging. Consequently, studies of linguistic structures produced during writing have often been based on the manual inspection of smaller data sets (cf. Section 2).

With the goal of alleviating this issue, we propose a methodology for extracting sentences from keystroke logs. Our approach is based on the notion of sentence histories: the collection of all the versions of a given sentence throughout the production of a text. We believe that sentence histories are the appropriate object of study for sentence production because they capture the complete evolution of a sentence.

The contributions of this work are threefold: we present a theoretical framework for extracting sentence histories from keystroke logs; we describe how this framework was implemented in a tool for automatic sentence history extraction; and we perform an evaluation of this tool on a sample of keystroke logs.

Our theoretical framework is derived from two novel approaches developed in parallel by two research groups in 2021. Its implementation is an open-source software which allows for the automated generation of sentence histories from keystroke logs and can be applied to any language for which automated sentence segmentation tools are available. We evaluate the framework's applicability and investigate challenges and limitations related to its implementation by generating sentence histories for a set of keystroke logging data in French, English, and German. The evaluation allows us to draw conclusions about the current capacities of the tool and provides pointers for further development of the concept of sentence histories and their automated generation and annotation.

Table 1. Adapted excerpt from a keystroke log from the evaluation corpus (Section 5.1). **Start_time** and **end_time**: millisecond from the beginning of the writing session at which the beginning and the end of the event occurred; **position**: position in text at which the event occurred; ␣: space character; ←: deletion.

ID	Start_Time	End_Time	Event	Position
1	870,085	870,176	m	445
2	870,245	870,329	i	446
3	870,466	870,546	e	447
4	870,608	870,684	u	448
5	870,724	870,819	x	449
6	870,822	870,888	␣	450
7	871,221	871,317	c	451
8	871,363	871,453	o	452
9	871,508	871,595	n	453
10	871,649	871,744	n	454
11	871,776	871,869	u	455
12	871,896	871,984	e	456
13	872,009	872,100	␣	457
14	872,574	872,644	s	458
15	872,662	872,791	o	459
16	872,740	872,844	u	460
17	872,851	872,938	s	461
18	872,946	873,051	␣	462
19	873,087	873,171	l	463
20	873,185	873,272	e	464
21	873,338	873,427	␣	465
22	873,708	873,798	n	466
23	873,842	873,964	o	467
24	873,952	874,038	m	468
25	874,531	874,614	␣	469
26	879,551	879,628	←	449
27	879,686	879,769	←	448
28	879,834	879,913	←	447
29	879,973	880,060	←	446
30	880,150	880,220	←	445
31	880,955	881,045	a	445
32	881,096	881,194	u	446
33	881,227	881,307	s	447
34	881,368	881,448	s	448
35	881,470	881,548	i	449
36	883,293	883,387	d	470
37	883,449	883,533	e	471
38	883,547	883,631	␣	472

The remainder of this paper is structured as follows: In Section 2, we present related work. Section 3 introduces fundamental concepts of our framework, and we provide details on the implementation of the framework in Section 4. Finally, we present an evaluation of the framework and its implementation (Section 5) and conclude the paper with a discussion of encountered challenges, observed limitations and possible ways forward (Section 6).

2. Related Work

Keystroke logs have been extensively used in the psycholinguistics of writing. In this domain, an important body of work focuses on the role of pauses in the writing process (e.g., Alves et al. 2007; Foulin 1995; Matsuhashi 1981), and there is evidence for pauses being cognitively motivated (Olive 2012). Keystroke logs, with the precisely recorded temporal information about each writing event, are particularly well suited to this research angle, and the existing works point towards a relationship between the pause distribution and duration on the one hand, and the linguistic content that is being produced on the other (Immonen and Mäkisalo 2017; Medimorec and Risko 2017). Pauses can also be seen as segmenting the data flux into units, which are referred to as *bursts* (Chenoweth and Hayes 2001). But operationalising bursts is directly dependent on the minimum pause length considered as a burst boundary. This effectively means that choosing a different threshold can produce widely different segmentations of the same text. And since this segmentation of the writing events happens in *chronological* order, the *linear* structure of bursts—and therefore their linguistic content—is not straightforward to apprehend (see Table 1 for an example).

Nevertheless, there have been attempts to establish a mapping between production data in the form of bursts and linguistic structure. Cislaru and Olive (2018) manually examine syntactic properties of bursts. This examination clearly shows that bursts do not coincide with traditionally recognised syntactic structures. The bursts they observe are highly heterogeneous and >50% are syntactically incomplete (e.g., they end in a preposition or in a determiner). Some of these incomplete structures are identified as recurrent and potentially having specific functions in the writing process. The authors also examine how syntactically incomplete bursts achieve completeness. Based on their findings, the authors argue for the status of bursts as units of linguistic production (as opposed to units of linguistic reception). Gilquin (2020) examines bursts through the lens of Construction Grammar and finds that some bursts correspond to units traditionally recognised as constructions. Using robust statistical analysis, Feltgen et al. (2022) and Feltgen et al. (2023) show that there is a relationship between the linguistic content being produced and the segmentation of the writing flux into bursts.

Note, however, that Cislaru and Olive (2018) and Gilquin (2020) base their work on a manual examination of a relatively small set of data. Feltgen et al. (2022) and Feltgen et al. (2023) focus on phenomena that are easily represented by wordlists (the conjunction *et* 'and' and the subject clitics in French, respectively) and therefore relatively simple to track in a corpus. All these works also call upon some form of manual annotation in order to complete their analysis. In order to facilitate large-scale data analysis, an automated process capable of identifying elements of linguistic structure within the production data flow is required.

Recent developments of the Inputlog software are a step in this direction. Leijten et al. (2015) and Leijten et al. (2019) aggregate the logged process data from the keystroke level to the word level and present a module for analysing writing process data with Natural Language Processing (NLP) tools. Note that the NLP analysis used by Leijten et al. (2019) is currently available only for English and Dutch. Furthermore, they track revisions at the word level only. While this type of information can shed important light on the writing process (see, e.g., the work of Serbina et al. (2017) on word class changes during production), we are interested in capturing a wider linguistic context based on which more complex evolution patterns can be studied. Lardilleux et al. (2013) propose a tool that allows for the alignment of segments of text throughout the writing process. However, their work is aimed at data visualisation rather than annotation. The first contributions to the analysis of the writing process from this perspective come from Mahlow et al. (2022) and Miletić et al. (2022), who base their respective works on the notion of *text history*.

Miletić et al. (2022) propose a methodology for semi-automatic keystroke log annotation which relies on reconstructing and annotating intermediate versions of each text. While this approach does provide a wider linguistic context, basing the methodology on the text as a unit of analysis makes it more difficult to identify evolution within individual sentences across different text versions.

Mahlow et al. (2022) present a first attempt to model the writing process at the sentence level. They introduce a novel concept of *sentence history*. First, they transform writing data into a text history: a list of all intermediate text versions. The individual text versions are then segmented into sentences, and different versions of each sentence are grouped together. A sentence history is a collection of all intermediate versions of a given sentence produced during writing.

The framework presented here is based on the notions discussed above, and hence, the two projects are presented in more detail in the following sections.

2.1. ProTEXT

The ProTEXT project[2] aims to shed light on cognitive mechanisms that underlie the writing process by combining information on linguistic structure of the text under production and the behavioural data recorded by the keystroke logs. To this end, the writing process data were on one hand *linguistically annotated*, and on the other hand, it was segmented into *writing bursts*, which is the basic unit of analysis for the writing process data that the project focused on. In the second step, in order to allow for a linguistic annotation of the identified bursts, the sentences under production were reconstructed.

Data annotation was conducted manually based on automatic preannotation. A pipeline was elaborated in order to integrate automatic preannotation and manual annotation in an optimised manner. It also allowed preserving the link between the linguistic annotation and the burst structure (Miletić et al. 2022).

The observations from the ProTEXT project hightlighted the need for three main improvements in the work on keystroke logs: a reliable approach for keystroke log preprocessing, an efficient pipeline for sentence reconstruction, and better automatic preannotation in order to dispense with or at least accelerate manual annotation.

2.2. THEtool

Mahlow et al. (2022) propose a method for modelling the writing process from a linguistic perspective using text and sentence histories. They developed a software application, THEtool[3], which transforms keystroke logs into text history and extracts sentence histories based on it. THEtool allows to track the evolution of each sentence produced during the writing process. It also provides information on what kind of edit operation has been made between subsequent sentence versions and based on this information offers a possibility to filter the revisions according to linguistic constraints.

The implementation of THEtool led to the discovery of certain challenges and limitations related to automatic modelling of writing based on keystroke logs. On one hand, the automatic parsing of the process data proved to be a challenging task due to differences in character encoding, discrepancies in the log structure produced by different logging tools, and high diversity of the writing behaviour among writers. On the other hand, the ill-formedness and incompleteness of the intermediate text and sentence versions also present a substantial difficulty and turn segmenting the character sequences retrieved from keystroke logs into sentence lists into a challenging task. As a result, correctly aggregating sentences into sentence histories is not always possible, as shown in Mahlow et al. (2022).

2.3. Present Work

The framework presented in this paper is an extension of the method and application presented by Mahlow et al. (2022) and its evaluation is based on the ProTEXT corpus presented in Miletić et al. (2022). The framework introduces a comprehensive theoretical foundation for the automatic modelling of writing on the sentence level which did not exist before and offers novel concepts for presenting and describing the sentence evolution. At the same time, its implementation leads to substantial improvement of the sentence segmentation quality, which in turn makes the generation of sentence histories much more reliable.

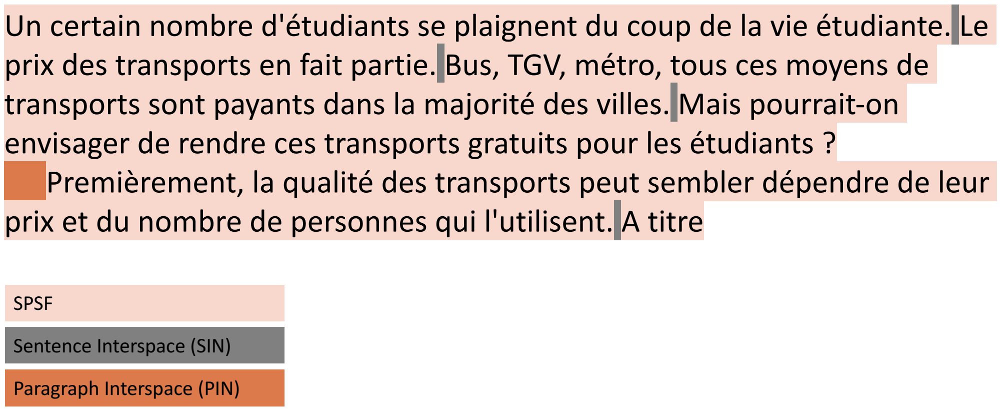
3. Theoretical Framework

3.1. Central Concepts and Terminology

We adopt and introduce a number of terms denoting objects of interest for the study of the writing process. Following Mahlow (2015), we represent the evolution of a given text as a series of text versions. A **text version** is a snapshot of the produced text at the moment in which a change in production mode occurs. A change in production mode is defined as the writer switching between (a) continuous writing at the edge of the text produced so far, (b) continuous deletion of existing content regardless of its position in the text, (c) continuous insertion of new content into the existing text (Mahlow 2015). For the content present in a given text version, we use the term **text produced so far** or **TPSF**. A collection of all the text versions for a given text constitutes a **text history** (Mahlow et al. 2022).

In an analogous manner, a **sentence version** is a version of a given sentence in a given text version. A collection of all sentence versions of a given sentence constitutes a **sentence history** (Mahlow et al. 2022). For denoting the content present in a sentence version, we introduce a new term inspired by the TPSF concept: **sentence produced so far** or **SPSF**.

In order to accurately track the content of each text version, we propose the concept of a **text unit** or **TU**. We consider that the full content of a given text version can be split into text units in such a way that each character produced, including whitespaces, belongs to one text unit. We distinguish between two main types of text units: SPSFs, which hold textual content, and interspace, which is used to separate SPSFs from each other and to structure the text. We distinguish between **sentence interspace (SIN)**, which is typically built of space characters and intervenes between sentence-level text units, and **paragraph interspace (PIN)**, which is typically comprised of newline characters and possibly indentation signalling the boundary between two paragraphs. An illustration is given in Figure 1.



Un certain nombre d'étudiants se plaignent du coup de la vie étudiante. Le prix des transports en fait partie. Bus, TGV, métro, tous ces moyens de transports sont payants dans la majorité des villes. Mais pourrait-on envisager de rendre ces transports gratuits pour les étudiants ?

Premièrement, la qualité des transports peut sembler dépendre de leur prix et du nombre de personnes qui l'utilisent. A titre

Figure 1. A visualisation of the text segmentation into text units: SPSF, SIN, and PIN.

3.2. Sentencehood Definitions in Literature

Attempts at defining a sentence and discussions around sentence definition have a long tradition starting back in antiquity (Cinato 2018). Some definitions primarily focus on formal (syntactic form), semantic (propositional content), or pragmatic (communicative function) aspects, while others combine multiple perspectives.

In the definitions of Bloomfield (1933), Allerton (1969), and Sauerland (2016), the main focus is on the syntactic form. For example, Bloomfield (1933) defines the sentence as a maximal independent linguistic form.

Noreen (1903), Bühler (1918), Wundt (1922), and Paul (2010) concentrate mainly on the semantic function. According to Bühler, sentences are "Sinneinheiten der Rede" ("meaning units of speech") (Bühler 1918, p. 18).

The pragmatic viewpoint is represented by Alston's definition, who sees the sentence as "the smallest linguistic unit that can be used to perform a complete action" (Alston 1964, p. 33).

Gardiner (1922) and Ries (1927) integrate multiple perspectives into one definition. Ries (1927) sees the sentence as "a grammatically constructed minimal unit of speech that expresses its content with regard to its relation to reality"[4]. The essence of the various formal, semantic, and pragmatic aspects found in traditional sentence definitions is well reflected in the summary of sentence attributes proposed by Panther and Köpcke (2008): "(1) Sentences are the maximal units of grammar. (2) Sentences relate to 'reality'. (3) Sentences have communicative potential" (Panther and Köpcke 2008, p. 88).

We consider all of these points of view as relevant when it comes to defining what a sentence is. However, these definitions consider the sentence in its final form. During the writing process, syntactic, semantic, and pragmatic properties of a sentence under production are in constant flux. Since our goal is to capture and describe this evolution, we require a means of marking the status of a given sentence version according to each of these dimensions. For this, we call on Matthews (1993), who discusses two defining criteria of a sentence: its completeness and its correctness, and proposes methods for testing these properties. Even though the author himself discusses the testability of sentencehood in a critical manner, we find that the notions of completeness and correctness are highly relevant for sentences under construction. As such, they constitute important concepts in our analysis.

3.2.1. Sentence Completeness

Looking back at the ancient perception of the sentence, we find Priscian's[5] syntactic theory with the idea of sentence completeness playing a central role. He focuses on the syntactic dimension of completeness and claims that a complete sentence results from an adequate arrangement of words (Fortes 2022; Matthews 1993). Noreen (1903), on the other hand, looks at the completeness from a semantic perspective and characterises a sentence as "capable of expressing a complete thought or meaning". Gardiner (1922) extends the notion of completeness further and argues that the sentence "always seems in a certain measure 'satisfactory'—satisfactory, that is to say, inasmuch as it is self-sufficient and complete psychologically and socially".

Based on different conceptions of completeness in literature, we distinguish in our framework between *conceptual* and *syntactic* completeness of a sentence and, additionally, introduce a criterion of *mechanical* completeness.

The **mechanical completeness** refers to the existence of a *sentence frame*: the capital letter at the beginning and the final punctuation mark at the end of a sentence[6]. The classification of a sentence as mechanically complete is based on its surface representation, which is the visible product of writing.

The definition of **conceptual completeness** is inspired by Gardiner's (1922) concept of satisfaction or psychological completeness with regard to sentences. We define the conceptual completeness of a sentence under production as a state when the writer seems to be satisfied with the sentence content. When working with keystroke logging data, we clearly do not know the writer's intention. Hence, when deciding on a sentence's conceptual completeness, we can only speculate. We base the speculations on writers' behavioural data: when the writer puts a final punctuation mark at the end of a word sequence and moves on with the writing process by producing or editing a different sentence (as opposed to revising the current sentence), we interpret their behaviour as a signal that they consider the sentence as complete.

The **syntactic completeness** is taken into consideration by Matthews (1993), inter alia, according to whom a "traditional test of completeness is that a sentence should contain a predication". Predication is understood here as the relation between the subject and the predicate. Although there are various possible syntactic representations of predication (Bowers 2001), we restrict our definition of syntactic completeness to the existence of a main clause with a lexical or pronominal subject and a predicate in the form of a finite verb form that agrees with the subject in person and number, which is inspired by the approach presented in Panther and Köpcke (2008).[7]

3.2.2. Sentence Correctness

Judging if a sentence is correct or incorrect, especially in terms of grammar, is not a trivial task. As Matthews argues: "grammarians tend to differ more about correctness than, in practice at least, about completeness" (Matthews 1993). We consider two dimensions of correctness: **mechanical correctness** and **grammatical correctness**.

In the case of written language, sentence production is typically regulated by prescriptive grammatical, orthographic, and typographic rules. Writers commonly intend to conform to them and readers tend to judge the text based on them. Hence, using a set of prescriptive rules to assess sentence correctness seems suitable in this context.

We classify a sentence as **mechanically correct** if it does not contain any punctuation, spelling, or capitalisation errors (Connors 1985).

With regard to grammatical correctness, we follow the principles of the correctness test discussed by Matthews (Matthews 1993): a **grammatically correct** sentence is a form which "should not be corrigible".

As we are working with sentences under production which often contain incomplete grammatical structures, typos, and/or incomplete words, we limit the grammatical correctness check to the sequence of mechanically correct words and exclude the remaining words. We also do not take into consideration the completeness of the grammatical structure, meaning a sentence can be classified as grammatically correct even if missing obligatory constituents (see Table 2 for examples).

Table 2. SPSFs with incomplete grammatical structures but building grammatically correct sentence versions according to our grammatical correctness definition. **GCOR** = grammatically correct.

SPSF	Gloss	GCOR
Alle diese Erfahrungen haben mich schlussendlich	All these experiences have finally	Yes
Immerhin verbringt man doch die meiste Zeit seines Lebens mit	After all, one spends most of one's life with	Yes
Und auch wir	And also we	Yes

3.3. *Sentence in the Writing Process Context*

From the text production perspective, Hayes sees sentences as being typically constructed "from proposed sentence parts in a complex activity involving idea generation, evaluation, planning, and reading the text produced so far" (Hayes 2009, p. 2). They are produced as "bursts of language intended for inclusion in the text" (Hayes 2009, p. 3). Sentence parts put together can build a complete and correct sentence, or a given sentence part can be removed even before being transformed into a sentence.

Tracking the evolution of sentences means collecting and classifying these "sentence parts" produced in various revisions. Each sentence part needs to be identified as belonging to one of the previously produced sentences (or sentence parts) or as being a fragment of a new sentence. This is one of the key challenges. When identifying the boundaries of the SPSFs, we cannot rely on the sentencehood criteria mentioned in the sentence definitions above. In most cases, SPSFs do not fulfil the sentencehood criteria related to morphosyntactic form, conceptual content or pragmatic function until the production process is finished. Additionally, an SPSF which can be identified as a sentence at the given moment can lose this property again in the subsequent revision process and transform into a single incomplete word or even a single letter. We describe our modelling of this phenomenon below.

3.3.1. Sentences and Sentence Candidates

Keystroke logs do not provide any information on the writer's mental representation of the sentence and therefore, when identifying and classifying SPSFs, we cannot take writer's intentions into consideration. The only information at our disposal when tracking the sentence production process with keystroke logging is the behavioural data (the keys the writer presses) and the text itself as the visible result of the writing process at the given moment. We use this information and make assumptions about the intended scope and nature of each SPSF.

In order to distinguish between full-fledged sentences and sequences of characters that do not meet the sentencehood criteria, we introduce two types of SPSFs: a **sentence (SEN)** and a **sentence candidate (SEC)**.

We interpret the writer's behaviour as follows: the writer indicates the beginning of a sentence by capitalising its first letter and indicates its end by entering a final punctuation mark (".", "?", or "!"). Following this interpretation, we define a **sentence (SEN)** as a sequence of characters that starts with a capital letter and ends with sentence-final punctuation.

Once a sequence of characters has been identified as a sentence, its status remains unchanged as long as the writer does not clearly signal a revision of the sentence scope by removing the capitalisation of the initial letter or adjusting the final punctuation mark. In other words, as long as the sentence frame stays untouched, we treat the sequence of characters within this frame as a sentence, even if other sentencehood criteria are not satisfied (see Table 3 for examples).

Table 3. Excerpt from a sentence history showing how an SPSF evolves from SEC to SEN and back and how it can be modified while still preserving the status of SEN.

SPSF	Gloss	SEN or SEC?
Meiner Meinung nach, bringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und förder	In my opinion, migrants bring a lot resources with them that should be used and promot	SEC
Meiner Meinung nach, bringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und fördern sollt.	In my opinion, migrants bring a lot resources with them that should be used and promote.	SEN
Meiner Meinung nach, bringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und fördern sollt	In my opinion, migrants bring a lot resources with them that should be used and promote	SEC
Meiner Meinung nach, bringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und fördern sollte.	In my opinion, migrants bring a lot resources with them that should be used and promoted.	SEN
Meiner Meinung nachbringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und fördern sollte.	In my opinionmigrants bring a lot resources with them that should be used and promoted.	SEN
Meiner Meinung nach bringen die Migrant:innen sehr viel Ressourcen mit, die man einsetzen und fördern sollte.	In my opinion migrants bring a lot resources with them that should be used and promoted.	SEN
Meiner Meinung nach bringen die Migrant:innen sehr vielRessourcen mit, die man einsetzen und fördern sollte.	In my opinion migrants bring a lotresources with them that should be used and promoted.	SEN
Meiner Meinung nach bringen die Migrant:innen sehr viele Ressourcen mit, die man einsetzen und fördern sollte.	In my opinion, migrants bring a lot of resources with them that should be used and promoted.	SEN

In contrast, a **sentence candidate (SEC)** is defined as a sequence of characters that does not start with a capital letter and/or does not end in sentence-final punctuation. In other words, it fails the mechanical completeness criterion. A sentence candidate can appear in different positions in a TPSF: (a) between the beginning and the edge of the text or (b) between a sentence interspace or paragraph interspace and the edge of the text or (c) between the beginning of the text and a sentence or a paragraph interspace or (d) between a sentence interspace or paragraph interspace and a sentence or (e) between two paragraph interspaces or (f) between two sentences. All of these positions are illustrated in Table 4.

Table 4. Possible positions of sentence candidates. *B* = beginning of TPSF, *E* = edge of TPSF. Bold highlights the sentence candidate considered. Last column indicates the total number of SPSFs in the example.

SEC Position	Example	SPSFs
B-**SEC**-E	***This is a story of a tortoise***	1
B-PIN-**SEC**-E	<tab> ***This is a story of a tortoise***	1
B-SEN-SIN-**SEC**-E	*This is a story of a tortoise.* ***The tortoise carries his home on his back***	2
B-**SEC**-SEN-E	***This is a story of a tortoise*** *The tortoise carries his home on his back.*	2
B-**SEC**-PIN-E	***This is a story of a tortoise*** <newline>	1
B-SEN-SIN-**SEC**-SEN-E	*This is a story of a tortoise.* ***The tortoise carries his home on his back*** *No matter how hard he tries he cannot leave home.*	3
B-SEN-PIN-**SEC**-SEN-E	*This is a story of a tortoise.* <newline> ***The tortoise carries his home on his back*** *No matter how hard he tries he cannot leave home.*	3
B-PIN-**SEC**-PIN-E	<tab> ***This is a story of a tortoise*** <newline>	1
B-SEN-**SEC**-SEN-E	*This is a story of a tortoise.****The tortoise carries his home on his back*** *No matter how hard he tries he cannot leave home.*	3

The definitions of what is and is not a sentence provided above present an obvious simplification of the sentencehood concept. However, since our framework is intended to enable the automated extraction of sentences, it is inevitably guided by the practical limitations of its implementation. The definitions need to be broad enough to encompass sequences ranging from a single letter to a morphosyntactically, conceptually, and pragmatically complete sentence and enable identifying SPSFs in the text under production in a systematic and reliable manner. The simplification allows us to collect all SPSFs per sentence and generate sentence histories, which is a prerequisite for the further analysis of sentences under production. This, in turn, makes it possible to apply additional automated tools and investigate the properties of each identified SPSF with regard to the remaining sentencehood criteria (see Section 3.3.2).

During the writing process, both a sentence and a sentence candidate can change status. A sentence candidate can be completed and turned into a sentence, or it can be merged with another sentence candidate or sentence. A sentence can also be revised so that it turns into a sentence candidate. We illustrate this in the example below.

This is a short text history containing seven text versions. Each text version contains SPSFs marked with letters *A*, *B*, and *C*. In text version 1, *A* and *B* are sentences, while *C* is a sentence candidate. First, the writer intends to merge SPSF (B) and SPSF (C) in revisions (2) to (5). SPSF (B) becomes a SEC in text version (2), and its status remains unchanged as long as the writer does not remove the capitalisation of the first letter in SPSF (C) in text version (4). This action is interpreted as a signal that the boundaries of the SPSFs are changing, and (B) and (C) now form a single SEC. In revision (7), the merged SPSF (B + C) is split again. The final punctuation entered after *back* in SPSF (B) indicates the intention of the writer to change the SPSF scope again.

(1)
1. (A) *This is a story of a tortoise.* | (B) ***The tortoise carries his home on his back.*** | (C) *No matter how hard he tries he cannot leave home*
2. (A) *This is a story of a tortoise.* | (B) ***The tortoise carries his home on his back*** | (C) *No matter how hard he tries he cannot leave home*
3. (A) *This is a story of a tortoise.* | (B) ***The tortoise carries his home on his back, so*** | (C) *No matter how hard he tries he cannot leave home*
4. (A) *This is a story of a tortoise.* | (B + C) ***The tortoise carries his home on his back, so o matter how hard he tries he cannot leave home***
5. (A) *This is a story of a tortoise.* | (B + C) ***The tortoise carries his home on his back, so no matter how hard he tries he cannot leave home***
6. (A) *This is a story of a tortoise.* | (B + C) ***The tortoise carries his home on his back so no matter how hard he tries he cannot leave home***
7. (A) *This is a story of a tortoise.* | (B) ***The tortoise carries his home on his back.*** | (C) *so no matter how hard he tries he cannot leave home*

3.3.2. Approximating the Degree of Sentencehood

An SPSF which is mechanically, syntactically, and conceptually complete and at the same time mechanically and grammatically correct is a prototypical sentence in our model. But an SPSF that does not meet any of these criteria can also occur in a text. Table 5 gives an overview of the properties of the sentences and sentence candidates with regard to the completeness and correctness criteria.

Table 5. Sentencehood criteria in relation to the type of SPSF: sentence or sentence candidate. MCOM = mechanical completeness, SCOM = syntactical completeness, CCOM = conceptual completeness, MCOR = mechanical correctness, GCOR = grammatical correctness.

SPSF	MCOM	SCOM	CCOM	MCOR	GCOR
SEN	+	+/−	+/−	+/−	+/−
SEC	−	+/−	−	−	+/−

As the criterion of mechanical completeness is the main distinction between the sentence and the sentence candidate in our framework, it is always fulfilled by the sentence and never met by the sentence candidate. Additionally, it is is a prerequisite for both the conceptual completeness and mechanical correctness. Hence, the latter two criteria can never by satisfied by sentence candidates. In the case of the remaining two categories, both the sentence and the sentence candidate can meet or miss them. The example in Table 6 shows how an SPSF can evolve from a sentence candidate not meeting any sentencehood criteria to a sentence with a maximum degree of sentencehood. The example in Table 7 shows how the sentencehood degree of a SEN can change as a result of revisions.

Each sentencehood criterion is investigated for each SPSF: it is tested if the given SPSF fulfils the criterion. If the criterion is met, the SPSF receives a score of 1 in the given category; otherwise, it receives a score of 0. This way, a total sentencehood score can be calculated: for example, if an SPSF fulfils all criteria, the score is 3/3 for completeness and 2/2 for correctness; if its content is just one letter not constituting a complete word, the score is 0/3 for completeness and 0/2 for correctness. Based on the scores, we distinguish between three degrees of sentencehood: full, partial, and missing. The partial sentencehood is further divided into three subcategories: (1) complete and incorrect, (2) partially complete and incorrect, and (3) partially complete and partially correct. A case that a sentence does not meet any completeness criteria but is correct with regard to mechanics and grammar is not possible due to the dependencies between these properties discussed above. An overview of the sentencehood degrees is presented in Table 8. Calculating the sentencehood degree for each SPSF allows for tracking the sentencehood evolution and provides a high-level view on the fluency and efficiency of the sentence production process.

Table 6. Sentence history containing 4 SPSFs (3 SECs and 1 SEN) meeting a different set of sentencehood criteria. MCOM = mechanical completeness, SCOM = syntactical completeness, CCOM = conceptual completeness, MCOR = mechanical correctness, GCOR = grammatical correctness.

	SPSF	**MCOM**	**SCOM**	**CCOM**	**MCOR**	**GCOR**
Ich s ('I s')	SEC	–	–	–	–	–
Ich stehe also in direkter Re ('So I stand in direct re')	SEC	–	+	–	–	–
Ich stehe also in direkter Relation zu zwei Sprachen ('So I stand in direct relation to two languages')	SEC	–	+	–	–	+
Ich stehe also in direkter Relation zu zwei Sprachen, zwei Ländern, zwei Kulturen und zwei Gesellschaften. ('So I stand in direct relation to two languages, two countries, two cultures, and two societies'.)	SEN	+	+	+	+	+

Table 7. Sentence history containing 4 SPSFs (1 SEC and 3 SENs) meeting a different set of sentencehood criteria. MCOM = mechanical completeness, SCOM = syntactical completeness, CCOM = conceptual completeness, MCOR = mechanical correctness, GCOR = grammatical correctness.

	SPSF	**MCOM**	**SCOM**	**CCOM**	**MCOR**	**GCOR**
Genauso wichtig ist es, wenn m ('It is just as important when y')	SEC	–	+	–	–	–
Genauso wichtig ist es, wenn man versucht zu verstehen, wie jemand anderes etwas versteht. ('It is just as important when you try to understand how someone else understands something'.)	SEN	+	+	+	+	+
Genauso wichtig ist es, wenn man versuchzu verstehen, wie jemand anderes etwas versteht. ('It is just as important when youtry to understand how someone else understands something'.)	SEN	+	+	+	–	–
Genauso wichtig ist es, wenn man versuchen zu verstehen, wie jemand anderes etwas versteht. ('It is just as important when you to try to understand how someone else understands something'.)	SEN	+	+	+	+	–
Genauso wichtig ist es, versuchen zu verstehen, wie jemand anderes etwas versteht. ('It is just as important to try to understand how someone else understands something'.)	SEN	+	+	+	+	+

Table 8. Possible sentencehood degrees.

Sentencehood Degree	Definition	Completeness Criteria	Correctness Criteria
Full	Complete and correct	3/3	2/2
Partial			
	Complete and incorrect	3/3	0/2
	Partially complete and incorrect	1/3 or 2/3	0/2
	Partially complete and partially correct	1/3 or 2/3	1/2
	Incomplete and correct	n/a	n/a
Missing	Incomplete and incorrect	0/3	0/2

To summarise, our framework is centred around the notion of sentence history: the collection of all the versions of a given sentence throughout the writing process. We define a sentence by its formal, morpho-syntactic, semantic, and pragmatic properties and consider completeness (mechanical, syntactical, and conceptual) and correctness (mechanical and grammatical) as sentencehood criteria. During the writing process, these properties of a sentence under production are in flux. In order to capture this evolution, we rely on the concept of a sentence produced so far (SPSF) while distinguishing between full sentences (SENs) and sentence candidates (SECs). Finally, we propose to capture the evolution of a sentence along different axes by evaluating the sentencehood of SPSFs according to the completeness and correctness categories mentioned above.

In the following section, we describe how this framework can be implemented.

4. Implementation of the Theoretical Framework

The framework implementation is a modification and an extension of the existing application THEtool[8], which is an open-source solution implemented in Python[9]. We extend THEtool with the following processing steps that allow for a more detailed and reliable analysis of sentences under production: (1) segmentation of text versions into text units and classification of text units based on their content, (2) aggregation of text units into sentence histories, and (3) evaluation of the degree of sentencehood for different sentence versions.

4.1. Automatic Segmentation of the Text into SPSFs

THEtool transforms the keystroke logs provided by a keystroke logging tool into a series of text versions. The keystroke logs are collected in an XML-based IDFX format which is a widely used format for storing keystroke-logging data (e.g., Inputlog in Leijten et al. (2012) and ScriptLog in Johansson et al. (2018)). The IDFX file contains detailed information about each keystroke detected during a writing session. Listing 1 shows two keystrokes logs recorded during the production of the English pronoun *it*.

Each text version extracted by THEtool from keystroke logs is a sequence of characters containing sentences, sentence candidates, sentence interspaces, and paragraph interspaces, but their boundaries are unknown at the beginning.

For sentence boundaries detection, we apply the statistical DependencyParser from spaCy, which is an open-source Python software library for advanced NLP (Montani et al. 2023). According to the spaCy technical documentation, the dependency parser provides the most accurate sentence boundaries from the tools offered by spaCy[10] and hence has been selected for our implementation. Our tests showed that it performs very well on complete sentences. However, in case of incomplete and ill-formed sentences, it does not always provide correct results, as already shown in Mahlow et al. (2022). An example of an erroneous segmentation encountered in our own data is presented in Table 9.

Listing 1. Example of raw logging data in XML format, showing the sequence needed to produce the pronoun *it*.

```
<event id="388" type="keyboard">
    <part type="wordlog">
        <position>567</position>
        <documentLength>569</documentLength>
        <replay>True</replay>
    </part>
    <part type="winlog">
        <startTime>233465118</startTime>
        <endTime>233465123</endTime>
        <key>VK_I</key>
        <value>i</value>
        <keyboardstate></keyboardstate>
    </part>
</event>
<event id="389" type="keyboard">
    <part type="wordlog">
        <position>568</position>
        <documentLength>570</documentLength>
        <replay>True</replay>
    </part>
    <part type="winlog">
        <startTime>233465771</startTime>
        <endTime>233465776</endTime>
        <key>VK_T</key>
        <value>t</value>
        <keyboardstate></keyboardstate>
    </part>
</event>
```

Table 9. Example of an incorrect automated detection of sentence boundaries by spaCy.

Unsegmented text:		*Diese Texte wurden auf Inception annotiert, indem die Fehler in den entsprechenden Fehlerkategorien markiert wur* ('These texts have been annotated on Inception by mar the errors in the appropriate error categories')
Segmented text:	(1)	*Diese Texte wurden auf Inception annotiert,* ('These texts have been annotated on Inception')
	(2)	*indem die Fehler in den entsprechenden Fehlerkategorien markiert wur* ('by mar the errors in the appropriate error categories')

In order to minimise the effect of these issues, the segmentation proposed by spaCy is improved by THEtool in two steps. First, each sequence identified as a sentence by spaCy is classified according to the four text unit categories: SEN (sentence), SEC (sentence candidate), SIN (sentence interspace), and PIN (paragraph interspace)[11]. The classification algorithm is an implementation of the definitions provided in Section 3.3.1 and is visualised in Figure 2.

The first processing step transforms each text version into a sequence of SENs, SECs, SINs, and PINs. The second step consists in verifying if the generated sequences are acceptable according to our definitions. We defined the following restrictions with regard to the text unit order:

- SIN can only follow SEN.
- SEC can follow PIN, SIN, or SEN. However, a sequence of SEN-SEC is assumed to be rare and occurs only if the writer does not enter a whitespace character between the text units.

- SEN can follow all text unit types: SEN, SIN, PIN, and SEC. However, a sequence of SEN-SEN is assumed to be rare and occurs only if the writer does not enter a whitespace character between the text units.
- SIN-SIN, PIN-PIN, PIN-SIN, SIN-PIN, SEC-SIN, and SEC-SEC sequences are not allowed. Sequences containing several whitespace characters are considered as a single interspace unit. SECs integrate any trailing space characters. Since SECs do not have any surface properties that would allow us to reliable identify their beginning and end, they are always considered to be delimited by text units of a different type. If we encounter any of the six unacceptable sequences of text units, we merge them into one text unit: SIN-SIN becomes one SIN, PIN-PIN turns into one PIN, both PIN-SIN and SIN-PIN are merged and build a PIN, and both SEC-SIN and SEC-SEC are transformed into an SEC.

In multiple cases, merging adjacent SPSFs of the same type leads to improving sentence segmentation provided by spaCy. If we look at the example of erroneous segmentation provided in Table 9, we see that for the sentences detected, there are two sentence candidates. Merging them by means of our algorithm leads to a creation of one sentence candidate. This, in turn, results in a correct segmentation. Merging may also cause a formation of a sentence if two sentence candidates combined together form a sequence which starts with a capital letter and ends with the final punctuation mark. The merging algorithm is presented in Figure 3.

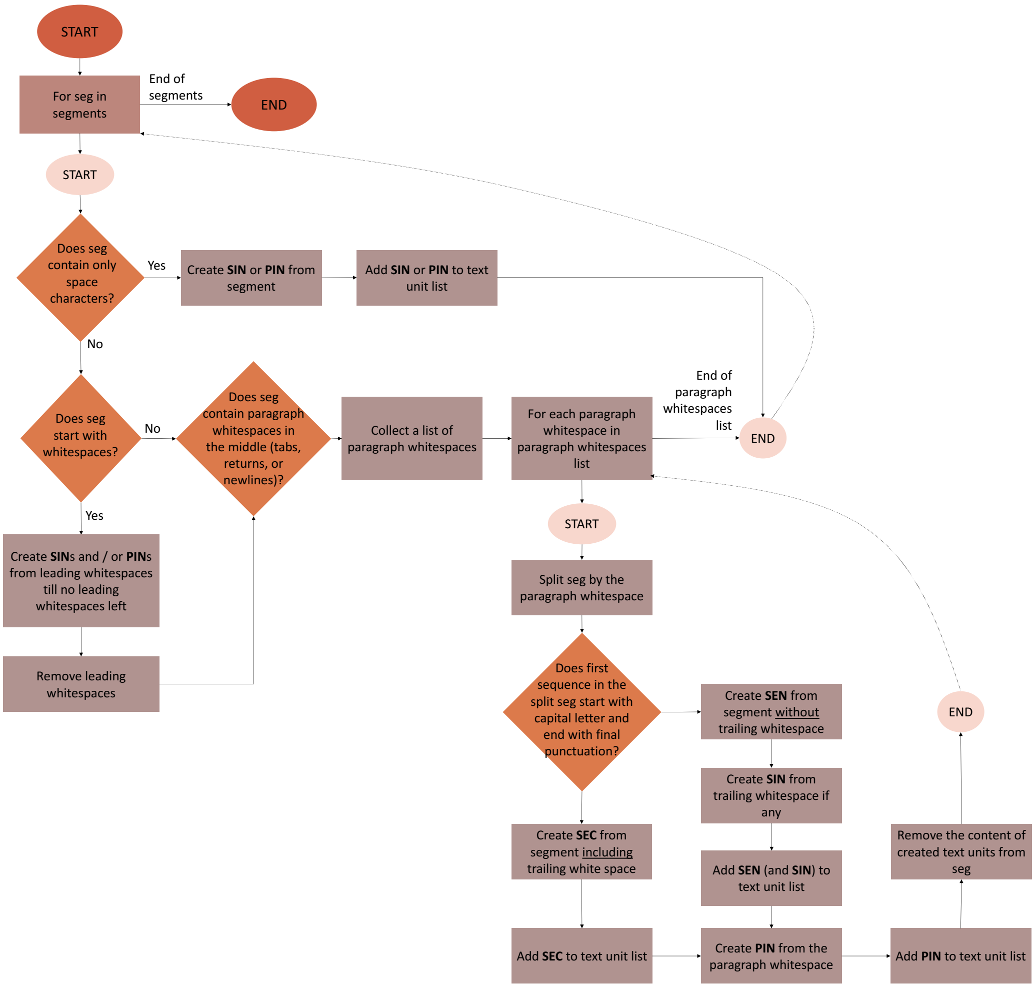

START
For seg in segments
End of segments
END
START
Does seg contain only space characters?
Yes
Create SIN or PIN from segment
Add SIN or PIN to text unit list
No
Does seg start with whitespaces?
No
Does seg contain paragraph whitespaces in the middle (tabs, returns, or newlines)?
Collect a list of paragraph whitespaces
For each paragraph whitespace in paragraph whitespaces list
End of paragraph whitespaces list
END
Yes
Create SINs and / or PINs from leading whitespaces till no leading whitespaces left
Remove leading whitespaces
START
Split seg by the paragraph whitespace
Does first sequence in the split seg start with capital letter and end with final punctuation?
Create SEN from segment without trailing whitespace
Create SIN from trailing whitespace if any
Add SEN (and SIN) to text unit list
Create SEC from segment including trailing white space
Add SEC to text unit list
Create PIN from the paragraph whitespace
Add PIN to text unit list
Remove the content of created text units from seg
END

Figure 2. Algorithm for assigning segments identified as sentences by spaCy to one of four text unit categories: SEN, SEC, SIN, or PIN.

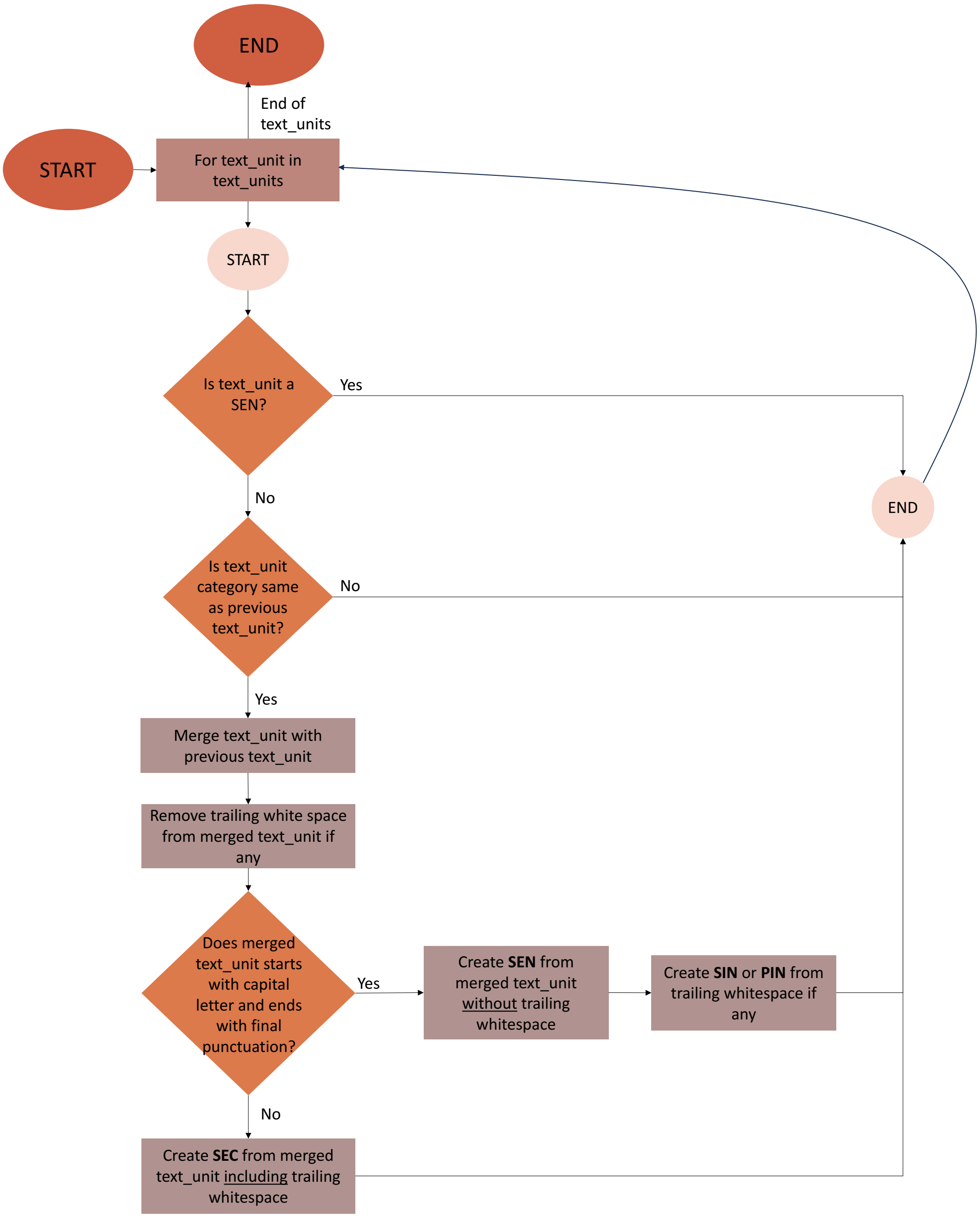

END
End of text_units
START
For text_unit in text_units
START
Is text_unit a SEN?
Yes
No
END
Is text_unit category same as previous text_unit?
No
Yes
Merge text_unit with previous text_unit
Remove trailing white space from merged text_unit if any
Does merged text_unit starts with capital letter and ends with final punctuation?
Yes
Create SEN from merged text_unit without trailing whitespace
Create SIN or PIN from trailing whitespace if any
No
Create SEC from merged text_unit including trailing whitespace

Figure 3. Algorithm for merging adjacent SECs, SINs, and PINs and transforming them into new SECs, SINs, PINs and SENs.

4.2. Automatic Aggregation of SPSFs into Sentence Histories

Once a given text version has been segmented into text units, an automated analysis is performed to determine which text unit has been impacted by the latest revision of the text and what kind of impact this was. Depending on the impact, a text unit can have one of the following states: (1) new, (2) modified, or (3) unchanged (refer to Figure 4 for a graphical representation of the algorithm). This classification is the basis for generating sentence histories. The sentence histories are extracted only from text units which belong to the category SPSF: sentences and sentence candidates. SINs and PINs are excluded from sentence histories. The sentence history generation is based on (1) the comparison of SPSF lists from the previous and the current text version and (2) the state of each SPSF in the current text version. The algorithm is presented in Figure 5.

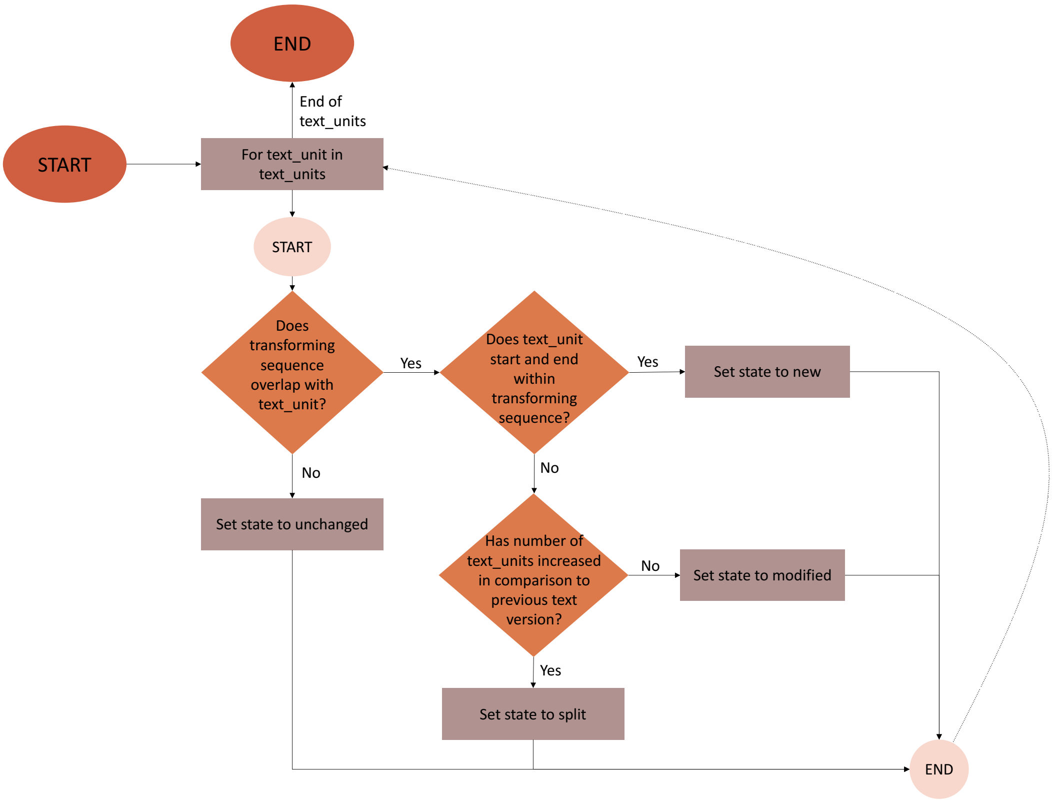

END
End of text_units
START
For text_unit in text_units
START
Does transforming sequence overlap with text_unit?
Yes
Does text_unit start and end within transforming sequence?
Yes
Set state to new
No
Set state to unchanged
No
Has number of text_units increased in comparison to previous text version?
No
Set state to modified
Yes
Set state to split
END

Figure 4. Algorithm for detecting TPSFs' states.

4.3. Measuring Sentencehood

The final processing step of our implementation performs an evaluation of the sentencehood criteria discussed in Sections 3.2.1 and 3.2.2. The tests are described below.

The mechanical completeness test consists in checking if the given sequence starts with a capital letter and ends with a final punctuation mark.

The conceptual completeness is dependent on the mechanical completeness of an SPSF. After the mechanical completion of an SPSF, it is verified where the next revision takes place. If the next revision does not impact the given SPSF, the SPSF is classified as conceptually complete.

For the syntactic completeness check, we apply an external Python library: spaCy (Montani et al. 2023). Each SPSF is parsed with the spaCy dependency parser. If an SPSF contains a subject in the form of a noun or a pronoun and a predicate in the form of a finite verb with the role of the root, it is labelled as syntactically complete.

For the correctness check, we apply a Python wrapper for an open-source proofreading tool called LanguageTool (Naber 2003). LanguageTool is based on human-curated rules for multiple languages. For the mechanical correctness check, it verifies if an SPSF contains spelling, punctuation or capitalisation errors. If no such errors are identified, we classify the SPSF as mechanically correct. We also check if the output from LanguageTool contains any grammatical errors. If no errors are detected, the SPSF is marked as grammatically correct. When checking correctness, typographic errors such as double whitespaces are not taken into consideration.

Thus, a sentencehood degree score can be calculated for each SPSF in a given sentence history (see Section 3.3.2 for details on the sentencehood degree score). Measuring the sentencehood degree of each SPSF allows for tracking the evolution of the sentence in a systematic and automated manner. For example, we can observe how the sentencehood of an SPSF increases and decreases as a result of revisions or how many SPSFs in the sentence history reached the full sentencehood degree at a certain point and lost it again. This enables us to collect statistics and draw conclusions related to sentence production efficiency but also helps to identify sentences or sentence parts which were particularly challenging for the writer. It can also allow for a better understanding of issues related to automated sentence processing with NLP tools by identifying the relation between sentencehood degree and quality of automated output.

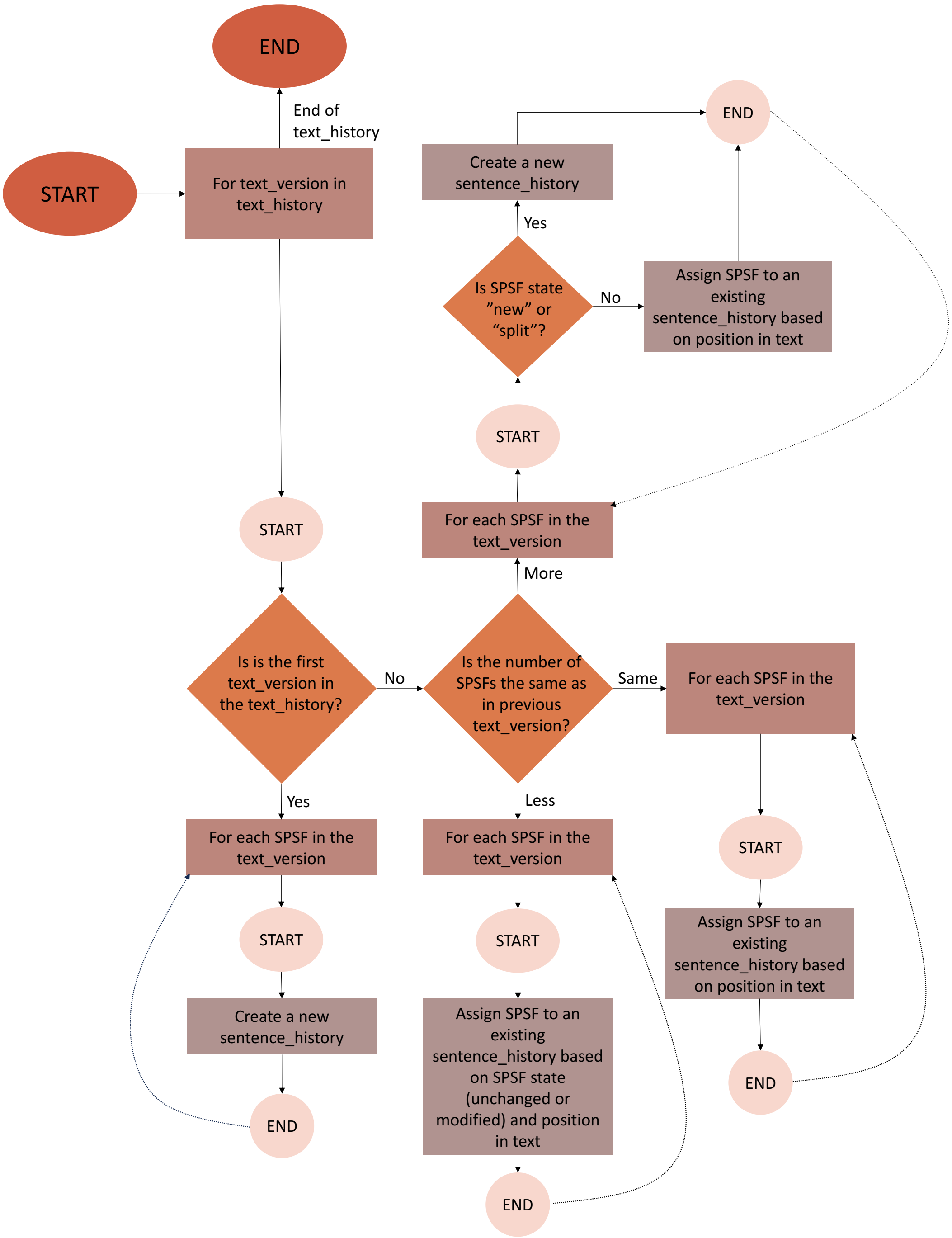

END
End of text_history
START
For text_version in text_history
START
Is is the first text_version in the text_history?
No
Yes
For each SPSF in the text_version
START
Create a new sentence_history
END
Is the number of SPSFs the same as in previous text_version?
More
Less
Same
For each SPSF in the text_version
START
Is SPSF state ”new” or “split”?
Yes
No
Create a new sentence_history
Assign SPSF to an existing sentence_history based on position in text
END
For each SPSF in the text_version
START
Assign SPSF to an existing sentence_history based on SPSF state (unchanged or modified) and position in text
END
For each SPSF in the text_version
START
Assign SPSF to an existing sentence_history based on position in text
END

Figure 5. Algorithm for generating sentence histories.

5. Evaluation of the Theoretical Framework and Its Implementation

In order to understand the applicability and reliability of the theoretical framework and its implementation in the context of writing process research, we performed a quantitative and qualitative evaluation of THEtool. This was achieved by processing keystroke logs from multiple writing sessions. The output was evaluated against a manually annotated gold standard corpus. The evaluation took into account the three processing steps described in Section 4: (1) segmenting the TPSF into text units and identifying text unit categories; (2) aggregating sentence histories; and (3) evaluating the sentencehood of the detected

SPSFs. In the remainder of this section, we first present the evaluation corpus and then describe in detail the three axes of evaluation.

5.1. Evaluation Corpus

The goal of this evaluation was to investigate how well the framework performs on texts with various properties produced in various conditions to deepen our understanding of its suitability for a wide range of scenarios. For this reason, we compiled a corpus of 10 texts written in three different languages (French, German, and English), belonging to different genres, produced by writers of varying ages, educational levels, and language competencies. The texts in French come from the Pro-TEXT corpus (Cislaru and Olive 2020), whereas the one in English and German were collected specifically for the evaluation of THEtool. The writing sessions were recorded with Inputlog (Leijten and Van Waes 2005) and ScriptLog (Johansson et al. 2018), and we used the IDFX files produced by these tools as input for THEtool (see Section 4.1 for more information on the IDFX format). An overview of the evaluation corpus is given in Table 10.

Table 10. Evaluation corpus.

Text ID	Lang.	Genre	Writer's Age	Education Level	Number of Keystrokes
Children_1	FR (L1)	essay	11–12	secondary school	551
Children_2	FR (L1)	narrative	8–9	primary school	334
Children_3	FR (L1)	narrative	8–9	primary school	552
Children_4	FR (L1)	narrative	10–11	primary school	583
Translation_1	FR (L1)	translation	young adult	university	3769
Composition_1	FR (L1)	essay	young adult	university	5012
Composition_2	FR (L1)	essay	young adult	university	2417
Composition_3	FR (L1)	essay	young adult	university	3326
Composition_4	DE (L1)	blog post	young adult	university	2710
Composition_5	EN (L2)	blog post	young adult	university	2653

We based the evaluation on text histories generated by THEtool. The quality of the extracted text histories was verified by comparing the final version of each text history with the corresponding final text generated by a keystroke logging tool. If no differences were detected, we assumed that the text history has been extracted correctly. This assumption is based on the observation that an error in a previous text version propagates to subsequent versions and in most cases impacts the extraction of the final text version. However, we cannot be sure that no errors occur in the extracted intermediate text versions. They have been neither verified nor corrected manually.

5.2. Evaluation of Text Unit Segmentation and Identification

This part of the evaluation focuses on two aspects: THEtool's capacity to correctly segment the text into text units and its capacity to assign them to the right text unit category (SIN, PIN, SEC or SEN). Additionally, to provide a wider context for this evaluation and demonstrate the encountered challenges, we investigate to what extent the concept of text units, as implemented at the moment, allows us to improve the initial sentence boundary detection performed by spaCy.

5.2.1. Evaluation Corpus

The segmentation and the identification of text units were evaluated on the corpus presented in Table 10. The manual annotation of the corpus was based on text histories extracted automatically for each text by THEtool. For each TPSF in a text history, the human annotator manually created a list of all text units constituting it (see Table 11 for an example). This file served as a reference annotation against which THEtool's output was evaluated.

Table 11. Excerpt from manual annotation of text units.

TPSF ID	Text Units	Total Text Units
0	SEN-SIN-SEN-SIN-SEN-SIN-SEN-SIN	8
1	SEN-SIN-SEN-SIN-SEN-SIN-SEN-SIN-SEC	9

5.2.2. Evaluation Metrics

We evaluate THEtool with respect to the segmentation accuracy and measure the ratio of over- and under-segmented TPSFs. Segmentation accuracy is calculated as the percentage of TPSFs in the given text history for which THEtool identified the correct number of text units. The over-segmentation error rate (OER) indicates the percentage of TPSFs that were segmented into more TUs compared to the manual annotation. Conversely, the under-segmentation error rate (UER) indicates the percentage of TPSFs that were segmented into fewer TUs compared to the manual annotation.

Additionally, for the texts with the highest and the lowest segmentation accuracy score, we compare the result of the SPSF boundary detection by THEtool with the initial sentence segmentation provided by spaCy.

When it comes to text unit identification, we compare the manual and the automatic annotation and measure the difference between the two sequences by calculating Levenshtein distance. Note that this is only performed on text histories for which THEtool has identified the correct number of text units. The content of the text units is not verified.

Filtering out incorrectly segmented TPSFs also allows us to disentangle the tasks and evaluate them separately. TU identification accuracy is calculated as the percentage of correctly classified TUs out of all TUs found in the correctly segmented TPSFs.

For the comparison of the SPSF detection performed by THEtool and by spaCy, we calculate how many of the SPSFs identified by the two tools are contained in manually created sentence histories. We concentrate merely on the textual content of the SPSFs and to this end remove all preceding and trailing whitespaces from all SPSFs.

5.2.3. Results and Discussion

We report TU segmentation accuracy and TU identification accuracy in Tables 12 and 13, respectively. The results are given for each text and as an average over the number of texts in the evaluation corpus.

The TU segmentation results show that THEtool's performance on this task are solid: 9 out of 10 texts were segmented with an accuracy >90%. It is also clear that it is more prone to under-segmenting than to over-segmenting given that only three texts have over-segmented TUs, whereas six have issues with under-segmentation.

Table 12. TU segmentation accuracy. # TPSF: number of TPSFs in text history. SEG acc.: TU segmentation accuracy (%). OER: % of over-segmented TPSFs. UER: % of under-segmented TPSFs.

Text ID	Genre	Age	# TPSF	SEG acc.	OER	UER
Children_1	essay	11–12	46	100.00	0.00	0.00
Children_2	essay	8–9	21	100.00	0.00	0.00
Children_3	narrative	8–9	21	95.24	0.00	4.76
Children_4	narrative	10–11	32	12.50	6.25	81.25
Composition_1	essay	young adult	235	98.30	1.70	0.00
Composition_2	essay	young adult	69	92.75	0.00	7.25
Composition_3	essay	young adult	254	98.82	0.79	0.39
Composition_4	blog post	young adult	100	100.00	0.00	0.00
Composition_5	blog post	young adult	217	99.54	0.00	0.46
Translation_1	translation	young adult	130	99.23	0.00	0.77
		Average	101	85.26	1.25	13.49

Table 13. TU identification accuracy. # corr. TPSF: number of correctly segmented TPSF used to calculate classification accuracy. # TU: total number of extracted TUs per text. CLASS acc.: classification accuracy (%).

Text ID	Genre	Age	# corr. TPSF	# TU	CLASS acc.
Children_1	essay	11-12	46	54	100.00
Children_2	essay	8-9	21	22	100.00
Children_3	narrative	8-9	21	23	100.00
Children_4	narrative	10-11	4	4	100.00
Composition_1	essay	young adult	231	4646	100.00
Composition_2	essay	young adult	64	823	100.00
Composition_3	essay	young adult	251	4868	100.00
Composition_4	blog post	young adult	100	2419	100.00
Composition_5	blog post	young adult	216	4744	100.00
Translation_1	translation	young adult	129	1886	100.00
		Average	103	1753.14	100.00

The TU identification results in Table 13 seem to indicate that this part of the task is trivial as long as the segmentation is correct. However, the segmentation itself is not trivial. It is noteworthy that there was one text in particular that proved difficult to segment: Children_4, for which the segmentation accuracy was only 12.5. Upon examining the data, we found that the main problem stemmed from sentence frame inconsistencies: as shown in Example 2, the first sentence ends in a space followed by a fullstop, which is followed immediately by a capital letter. This sentence boundary is not detected by THEtool.

(2) Il était une fois ,à l' école des enfants avec leurs amis qui joue au loup .Tout d,
‘Once upon a time ,at school children with their friends were playing tag .Sudd’

This specific situation could be considered as ambiguous, since the status of the text sequence after the fullstop is not clear at this stage. However, six text versions later, the second sentence is completed (see Example 3).

(3) Il était une fois ,à l' école des enfants avec leurs amis qui joue au loup .Tout d'un coup c'etait a shazya de touche et san faire espres elle a giffle morgane.
‘Once upon a time ,at school children with their friends were playing tag .Suddenly shazya was it and witout meanin to she slaped morgane’.

Now, the segment contains two sequences with an identifiable sentence frame (they start with a capital letter and end with a sentence-final punctuation). Nevertheless, this remains undetected, and the full sequence is considered as a single text unit. This error propagates through all the subsequent text versions, leading to a very low accuracy score.

In order to better identify the challenges related to segmentation and evaluate the performance of THEtool's algorithm for segmentation improvement (see Figure 3), we compared the SPSF detection accuracy of THEtool and spaCy for the texts with the worst and the best segmentation accuracy scores from the evaluation above. The results are presented in Table 14. Table 15 provides examples of segmentation errors found in the output of spaCy and/or THEtool.

As can be observed in Table 14, THEtool's algorithm manages to rectify some of the erroneous segmentations by spaCy (see the first three examples). However, the error in the last example (the one identified as the source of segmentation issues in Children_4) is not detected either by spaCy or by THEtool. This points towards the need to improve the robustness of both algorithms to sentence frame inconsistencies.

Table 14. SPSF detection accuracy of spaCy and THEtool on the example of 4 texts. SPSF DET acc.: SPSF detection accuracy (%)

Text ID	Genre	Age	SPSF DET acc. of SpaCy	SPSF DET acc. of THEtool
Children_1	essay	11–12	85.10	100.00
Children_2	essay	8–9	80.95	100.00
Children_4	narrative	10–11	56.41	79.49
Composition_4	blog post	young adult	100.00	100.00
		Average	80.62	94.87

Table 15. Example segmentation errors found in spaCy and THEtool output.

Text ID	SPSFs by SpaCy	SPSFs by THEtool	Correct SPSFs
Children_1	(SPSF 1) *Ce que je pense de la violence* (SPSF 2) *c’est que al* (SPSF 1) ‘What I think of violence’ (SPSF 2) ‘is that ats’	(SPSF 1) *Ce que je pense de la violence c’est que al* (SPSF 1) ‘What I think of violence is that ats’	(SPSF 1) *Ce que je pense de la violence c’est que al* (SPSF 1) ‘What I think of violence is that ats’
Children_2	(SPSF 1) *Je pance que la violance a l’ecole est un peut i peut tros danjereut il y a de la bagare de l’insuletans et* (SPSF 2) *de* (SPSF 1) ‘I think that violance at school is a litle bitt too danjerous there are fites insultings and ’ (SPSF 2) ‘de’	(SPSF 1) *Je pance que la violance a l’ecole est un peut i peut tros danjereut il y a de la bagare de l’insuletans et de* (SPSF 1) ‘I think that violance at school is a litle bitt too danjerous there are fites insultings and de’	(SPSF 1) *Je pance que la violance a l’ecole est un peut i peut tros danjereut il y a de la bagare de l’insuletans et de* (SPSF 1) ‘I think that violance at school is a litle bitt too danjerous there are fites insultings and de’
Children_4	(SPSF 1) *Elle vat giffler* (SPSF 2) *shazya puie ainsi de suite.* (SPSF 1) ‘She gos an slaps’ (SPSF 2) ‘shazya an so on’.	(SPSF 1) *Elle vat giffler shazya puie ainsi de suite.* (SPSF 1) Elle vat giffler shazya puie ainsi de suite.	(SPSF 1) *Elle vat giffler shazya puie ainsi de suite.* (SPSF 1) Elle vat giffler shazya puie ainsi de suite.
Children_4	(SPSF 1) *Il était une fois ,à l’ école des enfants avec leurs amis qui joue au loup .Tout d,* (SPSF 1) ‘Once upon a time ,at school children with their friends were playing tag .Sudd’	(SPSF 1) *Il était une fois ,à l’ école des enfants avec leurs amis qui joue au loup .Tout d,* (SPSF 1) ‘Once upon a time ,at school children with their friends were playing tag .Sudd’	(SPSF 1) *Il était une fois ,à l’ école des enfants avec leurs amis qui joue au loup .* (SPSF 2) *Tout d,* (SPSF 1) ‘Once upon a time ,at school children with their friends were playing tag .’ (SPSF 2) ‘Sudd’

5.3. Evaluation of Sentence History Aggregation

Another evaluation criterion is THEtool’s capacity to aggregate SPSFs into sentence histories, i.e., to identify which SPSFs represent different versions of the same sentence. We present the evaluation corpus, the metrics and the results below.

5.3.1. Evaluation Corpus

The aggregation of sentence histories is evaluated on the corpus presented in Table 10.

The annotation of the gold standard corpus consisted in manually building sentence histories. To facilitate the work of the human annotator, we use THEtool’s sentence history output as a preannotation, which is then verified against the manually identified SPSFs.

The manually annotated corpus contains, for each text, a collection of sentence histories. Each sentence history has a unique ID, and for each sentence version in the history, the file also indicates the position of the sentence version in the text as well as the sentence version itself (see Table 16 for an example).

Table 16. Excerpt from manual annotation of sentence histories.

Sentence History ID	TPSF ID	Position in Text	SPSF Text
101877	0	1	il etait une foi
101882	2	1	i
101886	4	1	Il etait ue
101886	5	1	Il etait u
101886	6	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copinee
101886	7	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine
101886	8	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et touta
101886	9	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout
101886	10	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout aje
101886	11	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a
101886	12	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a je mm
101886	13	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a je m
101886	14	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a je me suis retour
101886	15	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a
101886	16	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a cout je me suis retour nait et il y a un eleve qui ma tape la et qui ma pousse par terre.
101886	17	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a cout je me suis retour nait et il y a un eleve qui ma tape la et qui ma pousse par t
101886	18	1	Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a cout je me suis retour nait et il y a un eleve qui ma tape la et qui ma pousse par terre.

5.3.2. Evaluation Metrics

We frame the aggregation of sentence histories as a classification task. We consider each sentence history in the reference annotation as a class. For each SPSF in THEtool's output, we check if it was assigned to the right sentence history. We calculate the precision, recall, and F1 score for sentence aggregation.

Segmentation errors can lead to SPSFs in THEtool's output that do not exist in the reference file. All such SPSFs are considered as misclassifications.

5.3.3. Results and Discussion

We report sentence history aggregation metrics in Table 17. The metrics are reported on text level and as a macro-average over the whole evaluation corpus.

Table 17. Sentence history aggregation results. # senhis: number of sentence histories in reference corpus. P: precision. R: recall. F1: F1 score. Metrics are macro-averaged.

Text ID	**Genre**	**Age**	**# senhis**	**P**	**R**	**F1**
Children_1	essay	11–12	2	1.00	1.00	1.00
Children_2	essay	8–9	1	1.00	1.00	1.00
Children_3	narrative	8–9	7	0.71	0.70	0.71
Children_4	narrative	10–11	11	0.82	0.80	0.81
Composition_1	essay	young adult	28	1.00	1.00	1.00
Composition_2	essay	young adult	12	1.00	1.00	1.00
Composition_3	essay	young adult	18	1.00	1.00	1.00
Composition_4	blog post	young adult	30	0.94	0.94	0.94
Composition_5	blog post	young adult	24	0.92	0.92	0.92
Translation_1	translation	young adult	15	1.00	1.00	1.00
		Average	14.80	0.94	0.94	0.94

THEtool performs very well on this task: for six out of eight texts, it achieves perfect scores. This shows that THEtool's algorithms for detecting the state of an SPSF states based on the revision scope analysis and for aggregating the sentence history on this basis provide solid results and are well applicable for an automated analysis of writing data.

A closer look at the data underlines once again the importance of the correct segmentation. One of the problematic texts here is the one that proved most difficult in the TU segmentation task (Children_4). The reason for the low scores is the same unsegmented sequence of two sentences as in the SPSF detection evaluation: as the amalgamated sentence in THEtool's output is not correctly segmented, the second sentence is not recognised and the generation of its history is never triggered.

A similar problem occurs in the text with the lowest score (Children_5). Its text history contains the SPSF shown in Example 4. The fullstop is not recognised as a sentence boundary, and so the sequence remains identified as a single SPSF, which entails classification issues downstream.

(4) Il etait une fois cans on sortait de la classe j'ai jouer avec mait copine et tout a cout je me suis retour nait et il y a un eleve qui ma tape la et qui ma pousse par terre.n

'Once upon a time wen we were coming out of class I plaid with mai friend and sudenly I tur ned and one pupil hitme there and pushedme down.n '

These observations stress once again that any errors that occur in the initial sentence segmentation process tend to propagate and compromise the quality of subsequent processing steps.

5.4. Evaluation of Sentence Completeness and Sentence Correctness Detection

In Sections 3.2.1 and 3.2.2, we presented five sentencehood criteria: mechanical, syntactic, and conceptual completeness, and mechanical and grammatical correctness. For each SPSF, THEtool detects which of these criteria are met. Below, we evaluate the quality of the module for automatic sentencehood detection.

5.4.1. Evaluation Corpus

The detection of sentence completeness and correctness is evaluated on a corpus of 489 SPSFs selected from four blog posts written in German by young adults. Each SPSF was annotated manually with the five sentencehood criteria mentioned above. An example of the manual annotation in provided in Table 18.

Table 18. Excerpt from manual annotation of sentencehood. MCOM = mechanical completeness, SCOM = syntactic completeness, CCOM = conceptual completeness, MCOR = mechanical correctness, GCOR = grammatical correctness.

Sentence History ID	Sentence Text	MCOM	CCOM	SCOM	MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn ber* ('When I lok at my professional career so far')				MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn be* ('When I lo at my professional career so far')				MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn betrachte* ('When I look back at my professional career so far')				MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn betrachte muss ich feststellen, dass ich sehr vielseitige Br* ('When I look back at my professional career so far, I have to say that I very varied po')			SCOM	MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn betrachte muss ich feststellen, dass ich sehr vielseitige B* ('When I look back at my professional career so far, I have to say that I very varied p')			SCOM	MCOR	GCOR
146291	*Wenn ich meine bisherige berufliche Laufbahn betrachte muss ich feststellen, dass ich sehr vielseitige Berufserfahrungen machen durfte.* ('When I look at my professional career so far, I have to say that I have been able to gain very varied professional experience'.)	MCOM	CCOM	SCOM	MCOR	GCOR

5.4.2. Evaluation Metrics

We evaluate the sentencehood detection as a multilabel classification task. We consider each sentencehood property (MCOM, CCOM, SCOM, MCOR, and GCOR) as a class. An SPSF is assigned to one or multiple classes or remains unassigned. For each SPSF in THEtool's output, we check if the assignment was correct. We calculate the precision, recall, and F1 score for each of the sentencehood properties.

5.4.3. Results and Discussion

The evaluation results are presented in Table 19. THEtool detected the mechanical and conceptual completeness correctly for all sentences. It also performed very well on the syntactic completeness and grammatical correctness detection, where the scores are close to 1. The lowest scores are achieved in the mechanical correctness category.

Table 19. Sentencehood detection results. # occurrences: number of SPSFs which were assigned to the particular category. P: precision. R: recall. F1: F1 score. Metrics are macro-averaged.

Sentencehood Category	# occurrences	P	R	F1
Mechanical completeness	92	1.00	1.00	1.00
Syntactic completeness	378	0.98	0.99	0.98
Conceptual completeness	65	1.00	1.00	1.00
Mechanical correctness	59	0.86	0.81	0.83
Grammatical correctness	465	0.97	0.99	0.98
Average	211.8	0.96	0.96	0.96

Multiple errors in the syntactic completeness category result from the missing distinction between the main and the subordinate clauses in the current implementation of the framework. If the main clause of a sentence has not yet been produced at all, THEtool classifies the existence of the subject and the predicate in a subordinate clause as syntactic completeness. This does not correspond to our definition as provided in Section 3.2.1. However, if the beginning of the main clause has already been produced and it is missing a subject or a predicate, then the SPSF is correctly classified by THEtool as syntactically incomplete. The first three SPSFs in Table 20 provide examples for this observation.

When analysing the errors, we also discovered undesired behaviour related to subject and predicate detection. There are SPSFs where a single letter or an incomplete word at the subject position has been detected as a subject and similarly, an incomplete word at the predicate position has been recognised as a predicate. This is illustrated in the last four examples in Table 20. The early presumption about the existence of a subject or predicate seems to make the results less reliable. A potential solution to this would be to suspend the syntactic completeness judgment until the writer produces a complete word. This would allow us to asses more reliably if the SPSF turned into a syntactically complete sentence or not. A way to achieve this would be to remove the incomplete words from the given word sequence prior to sentencehood detection.

Table 20. Examples of errors in the syntactic completeness detection.

Sentence History ID	Sentence Text	SCOM (Manual)	SCOM (THEtool)
146291	*Wenn ich meine bisherige berufliche Laufbahn betrachte* (‘When I look at my professional career so far’)	No	Yes
221283	*Wenn Menschen in das Land migrieren* (‘When people migrate to the country’)	No	Yes
352745	*Jahre später, als ich mich schon in Verzweiflung wiegte, weil ich absolut nciht* (‘Years later, when I was already in despair because I absolutely could nto’)	No	No
300739	*Eigentlich ist e* (‘Actually i is’)	No	Yes
185864	*Natürlich waren die meixsten* (‘Of course mst were’)	No	Yes
352745	*Alle diese Erfahrungen hab* (‘All these experiences ha’)	No	Yes
300741	*Und auch wir wärn* (‘And we woud also’)	No	Yes

Another indicator for the potential need for preprocessing the SPSFs before assessing their sentencehood are errors discovered in the automated detection of grammatical correctness. Table 21 provides examples for differences between the human judgement which is based on the definitions provided in Section 3.2.2 and the automated detection by LanguageTool integrated in THEtool. The first sentence in the table is classified as grammatically correct despite the existence of the verb *gäbe* ‘would be’ in the first clause which obviously does not fit there. On the other hand, the second sentence in the table is classified as grammatically incorrect, although there are no grammatical errors in the sequence. There is just an incomplete word at the end.

Incomplete words, when detected in an SPSF, seem to typically lead to the detection of grammatical incorrectness by LanguageTool. This behaviour is not in line with our definition of grammatical correctness. On the other hand, when manually analysing the outputs of LanguageTool, we were also able to detect certain inconsistencies within this behaviour. As presented in the first example in Table 22, an incomplete word does not always lead to grammatical incorrectness. Both *wärn* ‘woud’ and *wär* ‘wou’ are incomplete words, but in the first case, the SPSF is classified as grammatically correct and in the second case as incorrect. At the moment, we cannot draw any clear conclusions about this behaviour, and additional tests and manual analysis are required to understand it better.

Table 21. Examples of differences in the detection of grammatical correctness between a human annotator and THEtool.

Sentence History ID	Sentence Text	GCOR (Manual)	GCOR (THEtool)
300739	*So habe ich nach anderen Möglichkeitn gäbe und bin auf den Studiengang 'Sprachliche Integration' gestossen.* ('So I looked for other optios would be and came across the degree programme 'Linguistic Integration'.')	No	Yes
185273	*Integration sowie die gesellschaftliche Partizipation geschieht zu einem seh* ('Integration and social participation happen at a ver')	Yes	No

Table 22. Examples of differences in the automated detection of grammatical correctness between SPSFs containing incomplete or incorrect words and SPSFs where all words are complete.

Sentence History ID	Sentence Text	GCOR (Manual)	GCOR (THEtool)
300741	*Und auch wir wärn* ('And we woud also')	No	Yes
300741	*Und auch wir wär* ('And we wou also')	No	No
185273	*Integration sowie die gesellschaftliche Partizipation geschieht zu einem seh* ('Integration and social participation happen at a ver')	Yes	No
185273	*Integration sowie die gesellschaftliche Partizipation geschieht zu einem* ('Integration and social participation happen at a')	Yes	Yes

The mechanical correctness is the category with the lowest scores. Our manual analysis showed that there are SPSFs which do not contain any mechanical errors but despite that are categorised as mechanically incorrect. On the other hand, we also observed SPSFs where punctuation went undetected. Table 23 provides examples of these issues. As with grammatical correctness, the reasons for this behaviour remain unexplained for now.

Table 23. Examples of errors in the automated detection of mechanical correctness.

Sentence History ID	Sentence Text	GCOR (Manual)	GCOR (THEtool)
185272	*Diese Ausgrenzung der Minderheitsgesellschaften kann zu einem grossen Teil verhindert werden, wenn sich ausländische Personen schnell integrieren.* ('This exclusion of minority communities can be prevented to a large extent if foreigners integrate quickly'.)	Yes	No
300740	*Ich kenne viele Menschen, die eine Arbeit haben die sie nicht erfüllt.* ('I know many people who have a job that does not fulfil them'.)	No	Yes

To sum up, our evaluation shows that the results of automated completeness detection in all three categories are solid. The inconsistencies in the current implementation of grammatical correctness detection deserve further attention; despite this, the results in this category remain reliable. On the other hand, the automated detection of mechanical correctness requires further refinement in order to provide solid results.

6. Discussion

The evaluation presented in Section 5 allowed us to have a better understanding of the applicability of THEtool on writing data and the scope covered by the framework and the software. It also helped us identify current limitations of both the framework and the tool. We discuss these aspects below.

6.1. Framework's Applicability and Performance

The evaluation results presented above show that both the theoretical framework for sentence history extraction and its implementation are operational. In its current state, THEtool is capable of taking as input a keystroke log file in XML format and outputting the list of sentence histories for the full text history, in which every sentence version is accompanied by an indication of its sentencehood degree. Furthermore, THEtool achieves solid results on all three steps of the process: segmentation and identification of text units, sentence history aggregation, and sentencehood evaluation. The module for text unit segmentation and identification is capable of identifying and correcting issues in the underlying segmentation performed by spaCy. THEtool's text unit algorithm can improve the SPSF extraction accuracy for up to 20% on a given text compared to spaCy. When it comes to sentence history aggregation, 6 out of 10 analysed texts are processed without errors, and only 2 out of 10 receive scores <0.9. THEtool's modules for evaluating mechanical completeness, syntactic completeness, conceptual completeness, and grammatical correctness are highly reliable with evaluation scores of $\geq$0.97. Only the mechanical correctness module scores lower.

THEtool's solid performance in the evaluation shows that its output is reliable and can be used as a foundation for further studies of the writing process. The sentence histories allow for a detailed observation of the sentence genesis. The sentencehood degree annotation can be used to track the evolution of the sentence along different axes (mechanical, morpho-syntactical, conceptual). Moreover, the reconstructed sentences, be they complete or not, can be fed to tools for automatic annotation, which would open avenues for different types of analyses.

The evaluation process also leads to interesting observations related to the writer's age and text genre. We have already noted that the most problematic texts for text unit segmentation and sentence history aggregation were often the ones written by young children. Among adult writing, manual inspection of sentence histories revealed a particularly relevant point about genres. The translation text in our evaluation corpus contains much fewer revisions around sentence boundaries. On the other hand, the texts that are original adult productions contain a noticeable amount of back and forth at this position. We can reasonably suppose that this is due to the fact that during translation, the writer is following the text unit structure of the original text and therefore hesitates less when making the decision about ending a sentence.

Given the current size of our evaluation corpus, the observations cited above remain anecdotal evidence. However, we believe that the existence of our framework will facilitate processing larger amounts of data, which will in turn allow for large-scale, robust analyses of such phenomena.

6.2. Challenges and Limitations

During our work, we have also faced challenges and identified limitations of the current version of the framework and its implementation in THEtool. We discuss a selection of them here.

Some of the issues that remain unresolved concern technical aspects of keystroke log processing and sentence extraction, one of them being the reconstruction of linguistic content from keystroke logs. The work presented here was conducted using logs from ScriptLog (Strömqvist and Malmsten 1998) and Inputlog (Leijten and Van Waes 2005). Comparing the XML outputs of the two tools revealed inconsistencies which made adaptations in THEtool's keystroke processing module necessary. Additional programming was also required in order to accommodate differences between operating systems or different keyboard layouts when it comes to key press combinations needed to produce certain special characters, especially for texts in French. An important amount of work was invested in order to overcome these difficulties and avoid producing incorrect sentence versions. Nevertheless, not all discovered issues could be resolved.

Another technical aspect that remains problematic for now was already touched upon in Sections 4 and 5: the suboptimal quality of the initial sentence segmentation by spaCy and its impact on the subsequent processing stages. Note nonetheless that segmenting a text containing incomplete sentences poses a challenge to sentence tokenizers, which, as a rule, are not trained on this type of data. One possible path for improvement in this regard would be to leverage sentence histories as training data for these tools.

We also discovered limitations related to sentencehood detection and a need to introduce at least two steps of sentence preprocessing prior to evaluating the sentencehood of an SPSF. The preprocessing should encompass extracting main clauses from the SPSFs. This would enable us to improve syntactic completeness detection and make it more consistent with our definition. Another preprocessing step should exclude incomplete words occurring in an SPSF from the syntactic completeness and grammatical correctness detection process. Incomplete words lead to errors in both categories. The two preprocessing mechanisms would allow for better alignment between the definitions, and the implementation and would make the results more reliable.

Another set of limitations that merit discussion are related to our framework itself. One of the constraints concerns the method for SPSF identification. The identification is based on the opposing notions of sentence and sentence candidate. At this processing stage, the distinction between the two is based on strictly formal properties: the sentence starts with a capital letter and ends with a sentence-final punctuation. All other strings of content characters are considered as sentence candidates. From the theoretical point of view, this definition of a sentence is oversimplified. We are well aware of this, as illustrated by our implementation of sentencehood evaluation which resides on more complex criteria. However, this initial simplification is driven by the absence of other data on which the TPSF segmentation can be based: since we do not have access to the writer's intentions, we can only make assumptions based on the behavioural data and the formal properties of the produced text. Moreover, our evaluation indicates that this approach yields good results and the output builds a solid basis for more advanced analysis.

We acknowledge that our definitions of sentencehood criteria and their implementations are simplified compared to the descriptions of these phenomena in the theoretical literature. For instance, the only check we perform for syntactic completeness includes verifying if the SPSF in question contains a finite verb and a realised subject which exhibit agreement. However, this definition does not preclude the presence of other elements, which may or may not be syntactically complete. Consider the following example: *He was in the*. Such an SPSF would be annotated as syntactically complete by our system, whereas it contains a clearly incomplete structure. Furthermore, the definition we adopt would prove ill adapted for null subject languages, which do not require an overtly realised subject. Once again, this limitation was dictated by the application scenario, which required definitions that could be applied in an automated manner and would be able to reflect the properties of sentences under production.

Another example of a limitation driven by this specific application setup concerns the definition of sentence histories. In the current state of the framework implementation, we consider that a sequence of characters belongs to the same sentence version as the previous one if at least one of the characters from the original sequence remains. If the initial sequence is fully deleted, the new sequence will be considered as the beginning of a new sentence history. This can be problematic in two ways.

First, consider the case in which a later sentence version reproduces the original one with a minor difference, such as in sentence versions 0 and 4 in Table 24.

Table 24. Sentence versions with minor differences.

Sentence Version	SPSF	Gloss
0	*il etait une foi*	'once upon a time'
1		
2	*i*	
3		
4	*Il etait ue*	'once upon an'

On the other hand, consider the case where characters from the initial sequence survive into the modified sequence, but the rest of the content is different (Table 25). The pronoun *il* in text versions 0 and 2 does not have the same syntactic or semantic properties: the former is an expletive subject that is considered to be semantically empty, whereas the latter is a personal pronoun with a referent. It can therefore be argued that the two text versions do not share the same content.

Table 25. Sentence versions with shared surface content and syntactic/semantic differences.

Sentence Version	SPSF	Gloss
0	*il etait une foi*	'once upon a time'
1	*il*	'he/it'
2	*il a décidé de*	'he decided that'

The first limitation could be addressed relatively straightforwardly. It could, for example, be defined that differences in capitalisation or whitespaces do not constitute a difference that triggers the creation of a new sentence history. The second is more delicate to handle. On the one hand, the syntactic parsing of incomplete sentences proves often incorrect (see Mahlow et al. 2022; Miletić et al. 2022); hence, it cannot be reliably used for detecting syntactically or semantically "path-breaking" revisions. Secondly, defining a "path-breaking" revision which triggers the generation of a new sentence history might prove highly challenging. One possible question that could arise when analysing the example from Table 25 is which sentence history the SPSF 1 (*il*) should belong to. Another question could refer to SPSF 2: is it another way of expressing the same thought or is it a new idea that arose in the writer's mind after producing the SPSF 0 and therefore an entirely new sentence? We choose to consider these questions as another justification of the approach we take: founding the decisions on the only reliable sources of information—the writer's behaviour and the text itself. Generating sentence histories in this manner provides us with a solid foundation for further processing. These further steps could, in the future, consist in creating more fine-grained sentence histories which result from further analysis or interpretation of the available data.

6.3. Outlook

As the evaluation showed, our theoretical framework and its current implementation build a solid foundation for the automated analysis of sentences under production. There are, however, areas which require improvement before our software can be used with very large volumes of writing data. These enhancements relate to extracting TPSFs from keystroke logs and improving the segmentation of TPSFs into text units.

The keystrokes processing module of THEtool needs to be more robust in order to be able to correctly extract sentence histories from any IDFX file produced by Inputlog or ScriptLog. This requires a detailed investigation of the differences between the outputs of these keystroke logging tools and inconsistencies within the outputs of one tool which result from differences in keyboard layouts or operating systems used in the writing sessions. The improvement of text segmentation into text units seems to be a much more straightforward task. The observations derived from the evaluation build a good basis for enhancements in this area.

The presented framework is a first attempt at a systematic analysis of sentences under production. As already stated, in the current version, it applies certain simplifications of sentence and sentencehood definitions. The simplifications built a necessary step for validating the concepts and understanding the data under investigation better. Having collected initial observations, we are now equipped to integrate new aspects into the analysis. One of them would be extending the scope of syntactic completeness evaluation or introducing appropriate SPSF preprocessing steps to enable more consistent and solid results. Another potential extension would be an algorithm for a more fine-grained assignment of SPSFs into sentence histories, including identifying "path-breaking" revisions which should trigger the generation of a new sentence history.

The framework and its implementation open new possibilities for analysing the writing data. Sentence histories allow for investigating revisions within one sentence. The sentence evolution can be analysed by identifying types of revisions: typo corrections, syntax-related changes, changes with semantic impact and more. At the moment, we do not take into consideration behavioural information relative to the duration of writing events, but measuring pauses within SPSFs and calculating production process duration per SPSF could build a first attempt to associate the linguistic information about SPSFs with assumptions about the writer's cognitive effort. Another potentially highly interesting avenue would consist of projecting writing bursts and transforming sequences (cf. Mahlow et al. 2022) onto SPSFs to better examine the relationship between syntactic structure and revisions.

We hope that our work on improving and extending THEtool's capacities contributes to the advancement of research on the writing process and opens new perspectives.

Author Contributions: M.A.U.: conceptualization, data curation, investigation, methodology, project administration, software development, writing—original draft, review and editing. A.M.: data curation, investigation, methodology, writing—original draft, review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: The work of Malgorzata Anna Ulasik has been supported by ZHAW DIZH Fellowship Call 2021. The work of Aleksandra Miletić has been partly supported by the Pro-TEXT project (grant number ANR-18-CE23-0024-01).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data used for evaluation will be made available for research purposes upon the publication of the paper at the following address: https://github.com/mulasik/wta, 31 January 2024.

Acknowledgments: We would like to thank Cerstin Mahlow and Michael Piotrowski for inspiring discussions, for reading drafts of the paper, and for their support. We would like to thank Ramona Peyer and Gina Vogel for the manual annotation of the German data. The authors would also like to thank the reviewers for their valuable feedback.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Notes

[1] This is obviously limited to the part of the process which is visible on the surface as text being produced.

[2] https://pro-text.huma-num.fr/, 31 January 2024.

[3] https://github.com/mulasik/wta, 31 January 2024.

[4] Original text: “Ein Satz ist eine grammatisch geformte kleinste Redeeinheit, die ihren Inhalt im Hinblick auf sein Verhältnis zur Wirklichkeit zum Ausdruck bringt”, (Ries 1931, p. 99) translated in (Panther and Köpcke 2008, p. 87).

[5] A Latin grammarian, author of the "Institutiones grammaticae", an exposition of Latin grammar, which is an influential work largely used in Europe in the Middle Ages.

[6] Note that this definition of the sentence frame is language-dependant. For instance, it would not be relevant for writing systems that do not distinguish between capital and small letters, such as Arabic.

[7] Note that this definition would be too restrictive when dealing with null subject languages.

[8] https://github.com/mulasik/wta, 31 January 2024

[9] https://www.python.org/, 31 January 2024

[10] https://spacy.io/usage/linguistic-features, 31 January 2024

[11] Please refer to Section 3.1 for the definitions of SIN and PIN and to Section 3.3.1 for the definitions of SEN and SEC.

References

Allerton, David J. 1969. The sentence as a linguistic unit. *Lingua* 22: 27–46. [CrossRef]

Alston, William P. 1964. *Philosophy of Language*. Englewood Cliffs : Prentice Hall. [CrossRef]

Alves, Rui Alexandre, São Luís Castro, Liliana de Sousa, and Sven Strömqvist. 2007. Influence of typing skill on pause–execution cycles in written composition. In *Writing and Cognition: Research and Applications*. Amsterdam: Elsevier , pp. 55–65. [CrossRef]

Bloomfield, Leonard. 1933. *Language*. New York: Henry Holt and Company. [CrossRef]

Bowers, John. 2001. Predication. In *The Handbook of Contemporary Syntactic Theory*. Edited by Mark Baltin and Chris Collins. Oxford: Blackwell, pp. 299–333. [CrossRef]

Bühler, Karl. 1918. Kritische Musterung der neuen Theorien des Satzes. *Indogermanisches Jahrbuch* 6: 1–20. [CrossRef]

Chenoweth, N. Ann, and John R. Hayes. 2001. Fluency in writing: Generating text in L1 and L2. *Written Communication* 18: 80–98. [CrossRef]

Cinato, Franck. 2018. *Ancient Greek and Latin Grammarians*. New York: Oxford University Press. [CrossRef]

Cislaru, Georgeta, and Thierry Olive. 2018. *Le processus de textualisation: Analyse des unités linguistiques de performance écrite*. Louvain-la-Neuve: De Boeck Supérieur. [CrossRef]

Cislaru, Georgeta, and Thierry Olive. 2020. French Keylog Writing Corpora. Available online: https://pro-text.huma-num.fr/ressources/ (accessed on 20 February 2024).

Connors, Robert J. 1985. Mechanical correctness as a focus in composition instruction. *College Composition and Communication* 36: 61–72. [CrossRef]

Feltgen, Quentin, Georgeta Cislaru, and Christophe Benzitoun. 2022. Étude linguistique et statistique des unités de performance écrite: Le cas de et. *SHS Web of Conferences* 138: 10001. [CrossRef]

Feltgen, Quentin, Florence Lefeuvre, and Dominique Legallois. 2023. Sujet clitique et dynamique de l'écrit: un éclairage par les jets textuels. *Discours. Revue de Linguistique, Psycholinguistique et informatique. A Journal of Linguistics, Psycholinguistics and Computational Linguistics* 32. [CrossRef]

Fortes, Fábio. 2022. "Order of Things" and "Order of Words" in Priscian: Grammatical and Logical-Ontological Arguments in the Division and Ordering of Word Classes in *De Constructione* (ars gl 2, 116.9–121.15). *Histoire Épistémologie Langage* 44: 139–54. [CrossRef]

Foulin, Jean-Noël. 1995. Pauses et débits: les indicateurs temporels de la production écrite. *L'année Psychologique* 95: 483–504. [CrossRef]

Gardiner, Alan Henderson. 1922. The definition of the word and the sentence. *British Journal of Psychology: General Section* 12: 352–61. [CrossRef]

Gilquin, Gaëtanelle. 2020. In search of constructions in writing process data. *Belgian Journal of Linguistics* 34: 99–109. [CrossRef]

Hayes, John R. 2009. From idea to text. In *The SAGE Handbook of Writing Development*. Edited by Roger Beard, Debra Myhill, Martin Nystrand and Jeni Riley London: SAGE Publications Ltd.

Montani, Ines, Matthew Honnibal, Adriane Boyd, Sofie Van Landeghem, and Henning Peters. 2023. explosion/spaCy: v3.7.2: Fixes for APIs and requirements (v3.7.2) . Zenodo. Available online: https://zenodo.org/records/10009823 (accessed on 31 January 2024). https://doi.org/10.5281/zenodo.1212303.

Immonen, Sini, and Jukka Mäkisalo. 2017. Pauses reflecting the processing of syntactic units in monolingual text production and translation. *HERMES—Journal of Language and Communication in Business* 23: 45–61. [CrossRef]

Johansson, Victoria, Johan Frid, and Åsa Wengelin. 2018. Scriptlog—An experimental keystroke logging tool. Paper presented at 1st Literacy Summit, European Literacy Network, Porto, Portugal, November 1–3.

Lardilleux, Adrien, Serge Fleury, and Georgeta Cislaru. 2013. Allongos: Longitudinal alignment for the genetic study of writers' drafts. Paper presented at International Conference on Intelligent Text Processing and Computational Linguistics, Samos, Greece, March 24–30 . Berlin and Heidelberg: Springer, pp. 537–48.

Leijten, Mariëlle, and Luuk Van Waes. 2005. *Inputlog: A Logging Tool for the Research of Writing Processes*. Technical Report Number 2005:11, Antwerp: University of Antwerp, Faculty of Business and Economics.

Leijten, Mariëlle, Eric Van Horenbeeck, and Luuk Van Waes. 2019. Analysing keystroke logging data from a linguistic perspective. In *Observing Writing*. Edited by Eva Lindgren and Kirk Sullivan. Leiden: Brill, pp. 71–95. [CrossRef]

Leijten, Mariëlle, Lieve Macken, Veronique Hoste, Eric Van Horenbeeck, and Luuk Van Waes. 2012. From character to word level: Enabling the linguistic analyses of inputlog process data. Paper presented at Second Workshop on Computational Linguistics and Writing (CL&W 2012): Linguistic and Cognitive Aspects of Document Creation and Document Engineering, Avignon, France, April 23. pp. 1–8.

Leijten, Mariëlle, Luuk Van Waes, and Eric Van Horenbeeck. 2015. Analyzing writing process data: A linguistic perspective. In *Writing(s) at the Crossroads: The Process-Product Interface*. Amsterdam and Philadelphia: John Benjamins, pp. 277–302. [CrossRef]

Mahlow, Cerstin. 2015. A definition of "version" for text production data and natural language document drafts. Paper presented at 3rd International Workshop on (Document) Changes: Modeling, Detection, Storage and Visualization, Lausanne, Switzerland, September 8. pp. 27–32. [CrossRef]

Mahlow, Cerstin, Malgorzata Anna Ulasik, and Don Tuggener. 2022. Extraction of transforming sequences and sentence histories from writing process data: a first step towards linguistic modeling of writing. *Reading and Writing (Online First)* 1–40. Available online: https://link.springer.com/article/10.1007/s11145-021-10234-6#citeas (accessed on 31 January 2024). [CrossRef]

Matsuhashi, Ann. 1981. Pausing and planning: The tempo of written discourse production. *Research in the Teaching of English* 15: 113–34.

Matthews, Peter. 1993. Central Concepts of Syntax. In *Syntax*. Edited by Joachim Jacobs, Arnim Von Stechow, Wolfgang Sternefeld and Theo Vennemann. Berlin: De Gruyter Mouton, pp. 89–117. [CrossRef]

Medimorec, Srdan, and Evan F. Risko. 2017. Pauses in written composition: On the importance of where writers pause. *Reading and Writing* 30: 1267–85. [CrossRef]

Miletić, Aleksandra, Christophe Benzitoun, Georgeta Cislaru, and Santiago Herrera-Yanez. 2022. Pro-text: An annotated corpus of keystroke logs. Paper presented at Thirteenth Language Resources and Evaluation Conference, Marseille, France, June 20–25. pp. 1732–39.

Naber, Daniel. 2003. LanguageTool: A Rule-Based Style and Grammar Checker. Master's Thesis, University of Bielefeld, Bielefeld, Germany.

Noreen, Adolf. 1903. *Vårt språk: Nysvensk grammatik i utförlig framställning. Bd 1*. Lund: C. W. K. Gleerup.

Olive, Thierry. 2012. *Writing and Working Memory: A Summary of Theories and of Findings*. New York: Psychology Press.

Panther, Klaus-Uwe, and Klaus-Michael Köpcke. 2008. A prototype approach to sentences and sentence types. *Annual Review of Cognitive Linguistics* 6: 83–112. [CrossRef]

Paul, Hermann. 2010. *Prinzipien der Sprachgeschichte*. Berlin: De Gruyter, vol. 6. [CrossRef]

Ries, John. 1927. *Beiträge zur Grundlegung der Syntax...: Was ist Syntax?* 2nd ed. Prague: Taussig & Taussig, vol. 1.

Ries, John. 1931. *Beiträge zur Grundlegung der Syntax...: Was ist ein Satz?* Prague: Taussig & Taussig, vol. 3.

Sauerland, Uli. 2016. On the definition of sentence. *Theoretical Linguistics* 42: 147–53. [CrossRef]

Serbina, Tatiana, Sven Hintzen, Paula Niemietz, and Stella Neumann. 2017. Changes of word class during translation – insights from a combined analysis of corpus, keystroke logging and eye-tracking data. In *Empirical Modelling of Translation and Interpreting*. Edited by Silvia Hansen-Schirra, Oliver Czulo and Sascha Hofmann. Berlin: Language Science Press, chp. 7, pp. 177–208. [CrossRef]

Strömqvist, Sven, and Lars Malmsten. 1998. ScriptLog Pro 1.04. In *User's Manual*. Göteborg: University of Göteborg.

Wundt, Wilhelm Max. 1922. *Die Sprache*. Stuttgart: Alfred Kroner, vol. 2.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.