Functional Design of LLM-based Human-AI Interaction

Von der Universtität Bayreuth
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

von

Florian Lehmann
aus Konstanz

1. Gutachter: Prof. Dr. Daniel Buschek
2. Gutachter: Prof. Dr. Florian Alt
3. Gutachter: Dr. Katy Gero

Tag der Einreichung: 08.12.2025
Tag des Kolloquiums: 02.07.2026

To my family.

Acknowledgements

First and foremost, I would like to thank Daniel Buschek for always providing meaningful advice throughout my PhD and for sharing his great HCI knowledge with me.

Special thanks to the Bayreuth Gang for the incredible times at conferences and university: Hai, Sven, Tim, Florian (F), Markus, Viktorija, Arthur, and Miroslav. To my fellow researchers Amy, Yong Ma, Francesco, Thomas, Linda, Flo (B), Changkun, Sarah (P), Rifat, Anastasiya, Markus (W), and Tony. I truly enjoyed sharing great times with you at IDCs, summer schools, and conferences.

I thank Anna Feit for our collaboration and for sharing her meaningful advice and support throughout this journey. Hendrik Heuer for our collaboration, the fun conversations, and for sharing his personal perspective on research. Lukas Mecke for the early brainstorming on LLM-supported writing and for the fun conversations at conferences. Thank you to Fiona Draxler, Anna Werner, Matthias Hoppe, and Robin Welsch for investigating the AI ghostwriting effect together. To Karim Benharrak for our collaboration on writer-defined AI personas and fun conversations at conferences. I also thank Andreas Komninos, Luis Leiva, Mark Dunlop, and Ioulia Simou for our collaboration on mobile text entry. A special thank you to Sarah Theres Völkel for her support and for sharing her unique insights into both research and industry.

I thank Nadine Rexfort for her unmatched management skills and the fun chats at university. To Krystsina, Jannek, Cola, and Niklas – thank you for working with me on your amazing theses. Your work has been truly inspiring, keep up your awesome work.

To my friends, thank you for the good times! I remember them vividly – especially the laughter, the play, and the sheer waste of colors. You kept me connected outside of academia. True legends, all of you.

I thank my parents and sisters for keeping me grounded and for being there for me my whole life. A big thank you also to my niece Maya, for teaching me the mindset of the younger generation.

Greatest thanks to Alice for her huge support while I worked on my PhD, and Linus our very best, you carry knowledge without a PhD: all you need is a stone and water, and fun is guaranteed.

Abstract

Interaction with large language models (LLMs) through conversational interfaces has achieved widespread adoption and gained popularity, as used in AI chat tools. These tools are commonly implemented as standalone applications and in separated sidebars. However, these common design choices pull users out of their workflows and add friction. Thus, the seamless integration of generative AI models into users' workflows and enabling effective human-AI interaction remain unresolved challenges.

Recent implementations of AI applications force users to interrupt their workflows and rely on written prompting to interact with AI, yet prompting adds cognitive demands, and context switches interrupt workflows. This thesis addresses these challenges through two primary research goals: (1) exploring the integration of generative models into workflows, and (2) understanding fundamental concepts in human-AI interaction, such as initiative and control.

Through functional prototyping of interactive systems integrating AI, this thesis examines empirically how control and initiative influence user experience and usability, along quantitative and qualitative dimensions, such as analyzing interaction logs and conducting questionnaire, showing that increased user control and a low system initiative foster authorship in AI-assisted writing. Furthermore, it statistically explores user motivations for using mobile keyboard suggestions, revealing strategies like completion and correction that prioritize user goals over efficiency. Methodologically, it introduces StudyAlign, an open-source framework for web-based studies, which has been validated in over ten experiments.

Conceptually, building on insights and reflections across the projects, this thesis introduces Functional Flexibility as a theoretical construct. This thinking tool enables researchers, designers, and engineers to analyze and reflect on how user interface (UI) implementations facilitate access to the functional space of large language models. This thesis advocates for *function-oriented design of human-AI interaction*, a constructive design approach for moving beyond conversational prompting interfaces and integrating AI capabilities closely into workflows to align with user goals. This research highlights the central role of UIs in shaping human-AI interaction and informs the future development of systems that prioritize functional design, enabling users to achieve their objectives effectively.

Zusammenfassung

Die Interaktion mit Large Language Models (LLMs) hat durch Conversational User Interfaces eine hohe Verbreitung und Popularität erlangt, vor allem durch KI-Chat-Tools. Solche Tools werden häufig als eigenständige Anwendungen oder in separaten Seitenleisten implementiert. Allerdings ziehen diese gängigen Designentscheidungen Nutzer aus ihren Arbeitsabläufen heraus. Die nahtlose Integration generativer KI-Modelle in die Arbeitsabläufe der Nutzer und die Ermöglichung einer effektiven Mensch-KI-Interaktion bleiben ungelöste Herausforderungen. Aktuelle Implementierungen von KI-Anwendungen zwingen Nutzer, ihre Arbeitsabläufe zu unterbrechen und sich auf schriftliche Prompts zu verlassen, um mit der KI zu interagieren. Doch das Formulieren von Prompts stellt kognitive Anforderungen und Kontextwechsel unterbrechen den Arbeitsfluss. Diese Arbeit adressiert diese Herausforderungen durch zwei zentrale Forschungsziele: (1) die Erforschung der Integration generativer Modelle in Arbeitsabläufe und (2) das Verständnis grundlegender Konzepte der Mensch-KI-Interaktion, wie Initiative und Kontrolle. Durch die Erstellung funktionaler Prototypen interaktiver KI-Systeme untersucht diese Arbeit empirisch, wie Kontrolle und Initiative die Nutzererfahrung beeinflussen – sowohl in quantitativer als auch in qualitativer Hinsicht. Dazu gehören beispielsweise die Analyse von Interaktionsdaten und Umfragen. Die Ergebnisse zeigen, dass eine erhöhte Nutzerkontrolle und reduzierte Systeminitiative das Autorengefühl beim KI-gestützten Schreiben fördern. Darüber hinaus werden die Beweggründe der Nutzer für die Verwendung von Wortvorschlägen von mobilen Tastaturen statistisch untersucht und Nutzerstrategien wie Vervollständigung und Korrektur aufgezeigt, welche die Nutzerziele anstelle der Effizienz priorisieren. Als methodische Grundlage wird StudyAlign eingeführt, ein Open-Source-Software-Framework zur Durchführung von webbasierten Experimenten, das in über zehn Experimenten validiert wurde. Konzeptionell aufbauend auf den Erkenntnissen und Reflexionen der Projekte führt diese Arbeit Functional Flexibility (funktionale Flexibilität) als theoretisches Konstrukt ein. Dieses Denkwerkzeug erlaubt es Forschern, Designern und Ingenieuren, zu analysieren und reflektieren, wie Benutzerschnittstellen den funktionalen Raum von LLMs zugänglich und nutzbar machen. Diese Arbeit schlägt ein funktionsorientiertes Design der Mensch-KI-Interaktion vor, ein konstruktiver Designansatz, der über dialogorientierte Schnittstellen hinausgeht und KI-Fähigkeiten eng in Arbeitsabläufe

integriert, um sie an den Zielen der Nutzer auszurichten. Diese Forschung unterstreicht die zentrale Rolle von Benutzeroberflächen beim Design der Interaktion zwischen Mensch und KI und bietet eine Grundlage für die zukünftige Entwicklung von Systemen, die funktionales Design priorisieren und es Nutzern ermöglichen, ihre Ziele effektiv zu erreichen.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Generative large language models (LLMs) such as generative pre-trained transformers (GPTs) [33, 10] recently gained popularity through implementations such as ChatGPT. These applications offer chat as a user interface (UI) to let users converse with models. People use these chat applications in everyday life, for example, to ask for advice, solve problems, and as writing assistants [12]. To reach their goals, users input text to instruct the model to fulfill a certain task. This input method is called *prompting* and actively explored in human-computer interaction (HCI) research (e.g. [35, 26, 15, 39]). Prompting is a popular input method for steering LLMs and is implemented in many solutions from industry, such as Microsoft's CoPilot, OpenAI's ChatGPT, and Claude. However, these features are implemented as stand-alone solutions separated from the main working application, i.e. in sidebars or distinct online tools, also adding cognitive demand [41, 52], instead of integrating AI functions more tightly into workflows. Despite the popularity of AI chatbots and prompting, it remains a challenge to integrate generative models into workflows and to enable effective human-AI interaction. In this dissertation, I address this problem space through two primary research goals: 1) exploring the integration of generative models in workflows and 2) understanding fundamental concepts in human-AI interaction, such as initiative and control.

To explore human-AI interaction in these two directions, I have designed, implemented, and evaluated human-AI systems. I chose functional prototyping as a core method for investigating interaction with AI. I relied on an iterative design process for building functional interactive prototypes and evaluated them in mixed-methods user studies, collecting and analyzing data from surveys, interviews, questionnaires, and interaction logs. Concretely, I have explored the tight integration of generative text models into user workflows, moving beyond conventional AI chatbot implementations. First, I have built prototypes that implemented different levels of the fundamental concepts initiative and control to investigate their influence on steering AI functions for writing on mobile devices. I found that increasing

user control and initiative in designs for writing with AI leads to users perceiving themselves as authors, although they involved AI into writing [P3]. Second, I have built prototypes of an online text editor, offering a conversational UI, a toolbar UI, and free prompting to delegate tasks to AI [P1]. Experiments with these prototypes revealed that, interacting with the conversational UI, participants used short command-like task delegations, in contrast to the more elaborate instructions we had elicited in a survey before. The toolbar UI provided pre-specified AI tools and free prompting. People combined the tools for specific tasks and instructed AI via free prompting to realize custom functions. Along grounding the choice of AI functions in a particular context on fundamental concepts and evaluating functional prototypes, understanding users is also important for such functions. For example, in an analytical study, I have investigated user motivations for selecting word suggestions on mobile keyboards, and found that users follow different strategies for selecting word suggestions that go beyond efficiency, for example, completion, correction, and next-word prediction [P4]. This finding highlights that even in well-established interfaces with AI features, user-desired functions can be discovered that could be put into focus more explicitly in design.

Building on the experience from conducting experiments with functional prototypes across these projects, I have built StudyAlign as a methodological foundation for running experiments in our research group. StudyAlign is a software system for conducting web-based studies and evaluating interactive (functional) prototypes [P2]. In over ten studies, e.g. [17, 6, 14, 53, 54], the system has proven its feasibility.

One of these studies was the work on functional flexibility [P1]. In this overarching contribution of this thesis, I have explored how UIs shape the access to LLMs, and introduced a theoretical thinking tool on functional flexibility. This thinking tool enables researchers, designers, and engineers to analyze and reflect on how specific UI implementations support users in locating their desired goals within the functional space provided by LLMs.

This dissertation provides the following contributions to research on human-AI interaction:

1. Studies on the design, implementation, and evaluation of human-AI interaction, such as writing with AI on mobile devices, and writing text with AI in online text editors. These studies reveal design directions and technical possibilities of integrating AI tightly into workflows.

2. The software framework StudyAlign, an open-source project, serves as a methodological foundation for conducting web-based user studies with functional interactive prototypes.

3. Introducing Functional Flexibility as a fundamental concept that facilitates function-oriented design of human-AI interfaces. As a thinking tool, this concept allows assessing the analytical, critical, and constructive value of functional flexibility in human-AI interfaces.

Overall, this research highlights that the central role of UIs in human-AI interaction is to provide access to the functional space of AI, enabling users to achieve their goals and explore the functions offered by LLMs. The concept of Functional Flexibility informs the integration of AI into UIs and shapes how we will design the interaction with future human-AI systems. As a final outcome, this thesis introduces *function-oriented design of human-AI interaction*, an iterative design approach for creating AI functions to serve user intents.

Chapter 2

Contributing publications

This thesis is a cumulative dissertation. It builds on selected publications that I have published within my PhD. Together, these publications contribute to the bigger picture of my research. I was the first author on all of these publications, however, I have collaborated with other researchers on these works. Thus, the goal of this section is to 1) list the publications contributing to this thesis, and 2) make the authors' contributions transparent. Moreover, since all of these works were collaborations, I use the scientific "we" throughout the following chapters.

2.1 Publication list

The following publications contribute to this cumulative dissertation. Citations of these works have the prefix P to distinguish them from other related work.

[P1] Florian Lehmann and Daniel Buschek. Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts. arXiv:2410.10644 [cs]. 2024. DOI: 10.48550/arXiv.2410.10644.

[P2] Florian Lehmann and Daniel Buschek. StudyAlign: A Software System for Conducting Web-Based User Studies with Functional Interactive Prototypes. arXiv:2505.13046 [cs]. 2025. DOI: 10.1145/3733053.

[P3] Florian Lehmann et al. "Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship". In: *Mensch und Computer 2022*. ACM, 2022, pp. 192–208. DOI: 10.1145/3543758.3543947.

[P4] Florian Lehmann et al. “Typing Behavior is About More than Speed: Users’ Strategies for Choosing Word Suggestions Despite Slower Typing Rates”. In: *Proceedings of the ACM on Human-Computer Interaction* 7.MHCI (2023), pp. 1–26. DOI: 10.1145/3604276.

Special recognition: [P3] received an Honourable Mention Award.

2.2 Contribution statement

In Table 2.1, I provide an overview of the authors’ individual contributions to the publications.

Table 2.1 Contributing publications and the individual author’s contributions.

Publication	Author Contributions
Suggestion Lists vs. Continuous Generation [P3]	I came up with the idea for the interactions. I guided Niklas in the design and implementation of the prototypes together with Daniel and Hai. Niklas conducted the study according to my study design and analyzed the data under my guidance. I wrote the paper. Daniel edited the final version of the paper.
Typing Behavior is About More than Speed [P4]	I revisited the data analysis and wrote the paper. Anna guided Itto’s initial data analysis. Daniel and Anna edited the final version of the paper.
StudyAlign [P2]	I have designed and implemented the software system. Our working students Niklas and Jannek implemented the admin frontend under my guidance. I wrote the paper. Daniel edited the final version of the paper.
Functional Flexibility [P1]	I had the initial idea, designed and implemented the formative survey. Daniel, Hai, and Sven supported me with the coding of the survey responses. I designed and implemented the prototypes. Furthermore, I created the study design and conducted the studies. Moreover, I came up with the idea for our construct on functional flexibility. I analyzed qualitative and quantitative data, and wrote the paper. Daniel edited the final version of the paper and provided feedback throughout the project of four years.

Chapter 3

Background and key concepts

This section provides a brief overview of important aspects of this thesis, focusing mainly on the technical aspects and background in HCI and human-AI interaction. Furthermore, it provides definitions for reoccurring concepts in this thesis, such as initiative and control. This section is based on my contribution to the CHI doctoral consortium [24].

3.1 Technical background

Research on deep learning contributed to work on generative models. These models are the foundation for current implementations of human-AI interaction. In particular, these models are able to generate text based on a given input. Depending on the input text, these models cover various tasks, such as translating, question answering, and paraphrasing. Basic research to enable these generative models were Transformer networks by Vaswani et al. [44]. Different from existing encoder-decoder architectures in the past, recurrent layers were replaced by self-attention mechanisms. Based on these Transformer networks is the language model GPT-3 [10]. This model is the successor of GPT-2 [33]. With zero-shot and few-shot learning [10] approaches, their capability to extend text can be steered to fulfill certain tasks, for example, translating text from one language into another. GPT-3 was also the original model used in the initial release of ChatGPT. The model has been succeeded by several versions. This thesis focuses on making such generative models usable from a user perspective. As the main part of this thesis, we focused on building and investigating novel interaction techniques for generative AI models. In [P3], we relied on GPT-2 to enable writing of short messages with AI-generated phrase suggestions and automatic AI writing. In our work on functional flexibility [P1], we built the models OPUS-MT [42] (translating), T5 [34] (summarizing), GPT-Neo [9] (extending), and GPT-3 [10] (extending

and free prompting) into our prototypes of a collaborative online text editor to investigate how user interfaces shape the access to the functional space offered by LLMs.

3.2 HCI- and human-AI interaction background

Giving users control over model generations, e.g. through prompting interfaces, is a current theme in HCI research. There are efforts, for example, on making prompting explorable [38, 47], identifying challenges [52], exploring meta-cognitive demands [41], introducing prompt engineering toolkits [2], proposing design recommendations for envisioning prompts [39], and introducing a framework for prompting design spaces to explore LLM generated output [40]. Besides such efforts on prompting, HCI research investigates human-AI interaction from various perspectives. From a design perspective, work by Dove et al. [16] identified challenges of using machine learning models as a material for UX design. Further work in this direction was carried out by Yang et al. [48, 49]. In our work [P1], we approached human-AI interaction from a theoretical perspective, and suggest that specified tools are more suitable than prompts, e.g. for repeated tasks. In more specific work on language-based models, Yang et al. [50] explored sketching methods for NLP-enabled tools and features. These papers see models primarily as a design material and analyze challenges for designers building intelligent tools. Intelligent tools, in particular when offering agents, oftentimes provide mixed-initiative interaction [21]. In mixed-initiative interaction, the system can also take the initiative to do something, such as performing a task more actively. Horvitz [21] introduced 12 principles to take into consideration when building initiative into tools. In our contribution on writing with AI on mobile devices [P3], we suggest considering initiative as a design variable when designing mixed-initiative interfaces. More recently, Amershi et al. [1] and Weisz et al. [45] proposed design guidelines for human-AI interaction. Design practitioners can potentially include these guidelines in their design workflow. To further improve reflecting on and constructing specific UI implementations of generative AI applications, we contributed a thinking tool on Functional Flexibility [P1]. Beyond the design phase, another line of research in human-AI interaction focuses on developing and evaluating functional prototypes that integrate generative language models. Collaborative writing with language models in related work happens often through suggestions (e.g. [3, 4, 11, P3]). Other work investigates writing with AI in specific contexts, such as work by Gero et al. [18] investigating how language models can support science writing, and Singh et al. [37] how generated multimedia inspires writers. Another study by Yuan et al. [51] explored how writers might use generative models in the process of creative writing. Datasets revealing generative capabilities of GPT-3 [10] have been published by Lee et al. [23]. Our

research on interaction with LLM has explored word suggestion usage on mobile devices [P4]. We also built functional prototypes for investigating writing with AI on mobile devices [P3], and in a collaborative online text editor via a conversational UI, AI tools, and prompting [P1]. Moreover, we have created a software system for conducting web-based experiments with such functional interactive prototypes [P2].

3.3 Key terms

Several key terms reoccur throughout this thesis. They represent fundamental interaction concepts and methods. To ensure clarity and establish a shared understanding, I provide definitions for these terms below.

3.3.1 Initiative

With generative models in human-AI interaction, it becomes possible that systems do things that are human-like, for example, systems generating text that is similar to human-written text. It is also possible to attribute an active role to a system with such skills. An example of that is collaborative writing: the generative model could become an author that actively collaborates on a text similar to human co-authors. Such AI co-authors could then "take the initiative to edit the text". Thus, we define initiative as "the power to do something". If initiative switches between users and the system, it is known as mixed-initiative interaction [21].

3.3.2 Control

Control is a fundamental concept in HCI, oftentimes referred to as *agency* or *locus of control*. A recent literature survey by Bennett et al. [7] captures the different nuances and interpretations of the meaning surrounding *control*. Based on their categorization, we relate to control as the *experience* of feeling in control. In our contributions, users may report about their experience interacting with AI systems as "feeling in control over something", for example, an action, a piece of work, or an interaction.

3.3.3 Prompting

Prompting is an input technique to instruct generative models such as LLMs through any text input [35, 30]. Concretely, users tell the AI their needs, and the AI will react to the input. In products from industry, prompting is manifested in conversational UIs, allowing users to

input instructions within a conversation. In this thesis, we refer to prompting as the input technique, but also to prompting interfaces – UIs that enable users to input instructions.

Chapter 4

Research approach and methods

In this section, we provide an overview of our research approach and methods applied in the contributing publications.

4.1 Iterative design and functional prototyping

Compared to traditional design evaluation, integrating machine learning models into workflows requires a modified approach to designing and evaluating prototypes. Work by Yang et al. [49] pointed out challenges and method implications, for example, that prototyping for emerging AI capabilities requires functional interactive prototypes. For our work, we relied on an approach that combined iterative design and functional prototyping: we informed our designs through brainstorming sessions and surveys, and sketched alternatives of user interfaces and interaction flows. Early in the process, we switched to prototyping the most favored versions as web prototypes. These prototypes then served as a grounding material for discussions and were iterated through research-informed decisions, such as findings from early user studies.

We applied this approach extensively for our prototypes to research functional flexibility [P1] but also for investigating control and initiative as design parameters [P3].

4.2 Design workshops

For generating design alternatives, we set up workshops with members of our research group. Our colleagues participated with us in these workshops and contributed sketches and ideas. We discussed each design suggestion, considered them from different angles, and developed the ideas further. We ran a design workshop for [P1] in the early design stage.

4.3 Questionnaires

We applied a mixed-methods approach for collecting quantitative and qualitative data. To assess qualitative data from participants, we included questionnaires in our experiment procedures. We integrated standardized questionnaires such as the system usability scale (SUS), individual items asking for different aspects about interaction, asking for general feedback, and socio-demographic data. We conducted questionnaires as part of our contributions [P1, P3].

4.4 Surveys

A survey is a comprehensive questionnaire that enables broad and in-depth exploration of a given topic. Online surveys allow reaching a broad population and are rather easy to scale. We conducted an online survey to inform the design and implementation of our prototype in [P1].

4.5 Inductive coding

In general, coding is a method to structure and analyze qualitative responses, for example, collected in an open-ended survey. There are different approaches to coding to reveal the underlying meaning of participants' responses. We applied inductive coding for our research. With inductive coding, researchers break the responses into smaller samples and code these samples. These codes are then discussed, reflected on, and iterated. The responses can be coded for a set of categories on the same dataset. The result is an annotated dataset that allows for further interpretation. We applied inductive coding to analyze the elicited input phrases that people would delegate to AI as part of our formative survey in [P1].

4.6 Semi-structured interviews

Another way to collect qualitative data is a semi-structured interview. With semi-structured interviews, researchers can reveal more qualitative aspects and elaborate on users' views. Researchers create a list of topics and questions they go through in the interviews, but are flexible in the order of asking the questions and extending the interview based on the conversation with a participant dynamically. We conducted semi-structured interviews in our studies in [P1, P3].

4.7 Think aloud

Think-aloud protocols are a well-known method in HCI. Participants speak out loud what they think and do while they are performing the experiment task. Similar to think-aloud protocols, we asked participants to comment on their actions as part of the prototype usage in our studies [P1, P3].

4.8 Data analysis

We employed data visualization techniques and conducted a rigorous analysis of the collected data in our studies. We logged quantitative interaction data as part of our mixed-methods studies [P1, P3]. Our work on typing behavior [P4] comprises data visualization and statistical methods to examine user strategies in word suggestion usage.

4.9 Web-based experiments

Prototyping human-AI interaction using web technologies offers the advantage of conducting web-based experiments with them. These experiments can be conducted as controlled lab studies, but also as online field experiments. A field experiment combines features of lab studies and field studies. For example, the procedure is controlled and counterbalanced, while the study is open to anyone on the web. This approach allows researchers to maintain experimental control while optimizing the balance between internal and external validity. We conducted web-based experiments in our publications [P3, P1], and published StudyAlign, a software system for conducting web-based experiments [P2].

Chapter 5

Integrating generative models into human-AI workflows

The central research goals of this thesis are *to explore the integration of generative AI into workflows*, and *to understand fundamental concepts in human-AI interaction*, such as initiative and control. In particular, text-based models served as a material for the research in this thesis [P1, P3]. We have designed, implemented, and explored novel interaction methods integrating generative models in functional prototypes. These functional prototypes served as a basis to examine the fundamental concepts of user interaction with AI. I already laid out this scope in my contribution to the doctoral consortium at CHI'23 [24].

This chapter and the following chapter organize the original contributions into two areas. The contributions in this chapter focus on the research goal of *integrating generative models into human-AI workflows*. The contribution in the following chapter focuses on the research goal of *understanding fundamental concepts in human-AI interaction*, based on our concept of functional flexibility in generative AI interfaces. Within these chapters, I highlight the main findings we have identified in our published works and how they contribute to the bigger picture of effective human-AI interaction. The final section of the next chapter (Section 6.3) synthesizes the findings from our contributions into a *general approach to function-oriented design of human-AI interaction*.

5.1 Understanding user strategies for choosing word suggestions

This section is based on the publication

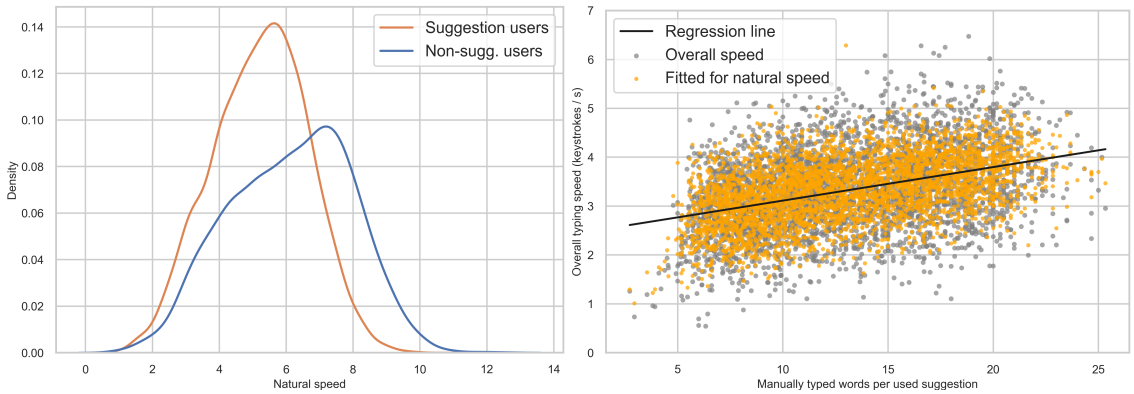
"Typing Behavior is About More than Speed: Users' Strategies for Choosing Word Suggestions Despite Slower Typing Rates" [P4].

To understand the user motivations for making use of system-generated text, we analyzed word suggestions on mobile keyboards. Suggestions on mobile keyboards is a well-established intelligent feature used by everybody writing on a smartphone. With a thorough data analysis in [P4], we investigated this intelligent feature to understand how users type with it and which strategies they employ when selecting system-generated word suggestions.

The analysis relied on data from existing work by Palin et al. [32]. We filtered their data and considered the typing behavior of 15,162 users. We computed four different typing speeds, such as the natural typing speed that excluded the use of word suggestions and only included keystrokes of lowercase letters that are preceded by lowercase letters, for common words that are shorter than five characters. Controlling for the natural typing speed, we showed that slower typists use suggestions more often and are slowed down by doing so, visualized in Figure 5.1. This user behavior motivated us to examine the user strategies for using word suggestions. In total, we identified eight strategies. The most often employed strategies were completion, correction, and next-word prediction. Based on these findings, we proposed a predictive model that includes these strategies for suggesting words in mobile keyboards.

This behavior shows that users follow strategies that eventually contradict one of the primary goals of keyboard suggestions, namely, to increase the writing speed. Similar to that, we found in [P1] that users typed requests to AI differently than assessed in a prior survey. These findings suggest that the intentions of system builders about usage and actual usage diverge. This observation highlights that it is important to understand users and their reasons for using intelligent features. We discuss in [P4] that not only usability but also models could be improved in this regard, for example, generative models could be fine-tuned for specific user strategies derived from actual feature usage.

These findings show that understanding real interactions and user reasons for making use of an intelligent feature has the potential to improve the design and function of such features.



(a) (b)

Fig. 5.1 Visualizations taken from [P4]. (a) Distributions of natural speed for *suggestion users* and *non-suggestion users*. (b) Effect of manually typed words per selected suggestion on typing speed: the more words are typed without suggestions, the faster the overall speed. Grey dots visualize the overall speed. Orange dots visualize the fitted values for overall speed, after controlling for natural speed.

5.2 Impact of control and initiative in interaction designs on perception of authorship

This section is based on the publication

"Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship" [P3].

To investigate control and initiative as fundamental concepts in human-AI interaction, we designed two interactive methods to write text with AI on mobile devices. With these methods, we manipulated the level of control and initiative offered to users. Concretely, the methods were *Writing with suggestions* and *Writing with continuously generated text*. With *Writing with suggestions*, participants composed text from phrases suggested by AI. With *Writing with continuously generated text*, AI continuously generated text, emulated as writing word-by-word with participants steering the writing. Screenshots of the interfaces can be seen in Figure 5.2. We also compared these methods to a baseline condition where users had to write manually, and found that writing with our two interaction methods made text longer and influenced wording. Furthermore, we discovered that adjusting control and initiative, and thus, altering the roles of the user and the system (writing and editing), also influenced perceived authorship. Our contribution [P3] was one of the first works uncovering the influence on perceived authorship when writing text with generative AI and also led to a later collaboration on this topic [17].

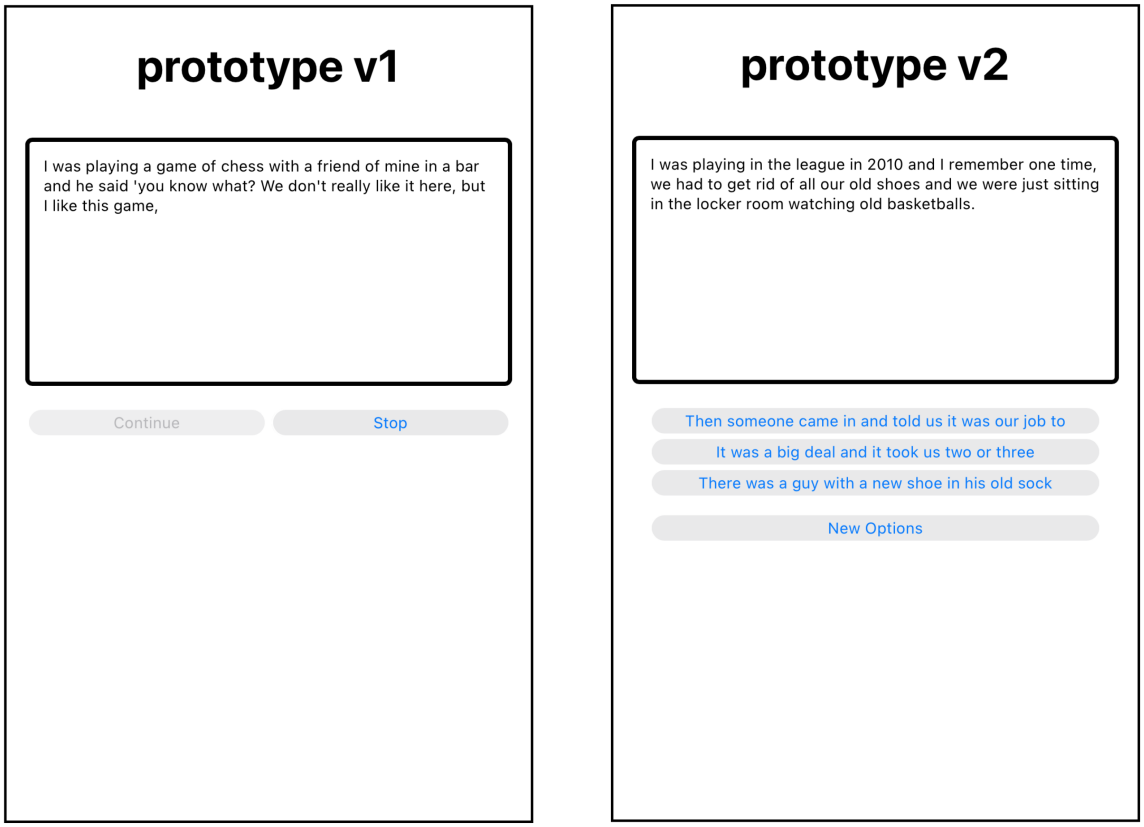


Fig. 5.2 Figure taken from [P3]. The final versions of the prototypes we implemented as mobile web applications. The prototype on the left shows the interface for writing with continuously generated text. The prototype on the right shows the interface for writing with suggestions.

In our original contribution, we conducted a within-subject study with 18 participants. We collected data on interaction, questionnaire responses, and interviews. Prototypes were implemented as web applications, and the generative backend relied on GPT-2.

The interaction methods differed mainly in the degree of implemented control and initiative: With the method *Continuous Generated Text*, the focus of the user shifts from writing to editing, and the system has the initiative, while user control was also reduced. In comparison, with the method *Writing with Suggestions*, the user had full control over adopting particular generated suggestions, and the initiative of writing the text remained with them. The data recorded in our study reflects these differences. Continuously generated text needs correction more often, and corrections are also longer, since the system-written text did not match user expectations, thus users had to take back initiative and exert control again. This user behavior is prevented when composing text with generated suggestions, since users decide upfront which suggestions to adopt. With our interaction methods, we provide

two practical examples of mixed-initiative interaction with generative AI. Furthermore, we challenge the implicit definition of initiative by Horvitz [21] (interfaces that enable efficient collaboration between agent and user), and suggest initiative as a design variable to explicitly manipulate and reason about when designing for mixed-initiative interfaces.

The analysis of the results revealed the main finding on perceived authorship. With both interaction methods, texts got longer, and people perceived that AI-generated text changed wording, compared to what people would normally write. However, users perceived themselves as authors when *Writing with Suggestions*, although user input was decreased the most with this method. By taking over suggestions for composing the texts, users also take over authorship. We see a reason for this perception in the high level of control as offered by this method, and users having the initiative in composing the text: The type of actions (editing actions vs. typing actions) is more important than just the number of actions. These actions result from the roles embedded in the interaction design. Thus, we suggest that control and initiative should be adopted as explicit design dimensions for designing roles as in frameworks such as [19, 31, 22].

In summary, these findings highlight the importance of interaction design decisions for human-AI interaction and how they impact the usability of AI functions and output.

5.3 Building functional interactive prototypes for investigating conversational AI, toolbars, and prompting

This section is based on the publication

"Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts" [P1].

To explore functional flexibility, we prototyped two versions of an interactive online text editor. These editors offered task delegation through interactive comments and a toolbar that offered pre-determined AI functions alongside free prompting of functions. In this section, we describe the design process of these text editors and present functional prototyping as a core method of our research.

LLMs offered new capabilities that were underexplored at the time when we started our long-term investigation of functional flexibility. Around the same time, related work identified challenges in designing human-AI interaction, such as sketching for NLP applications [50], and uncertainty about the outputs of the models as well as thinking about their abilities [48,

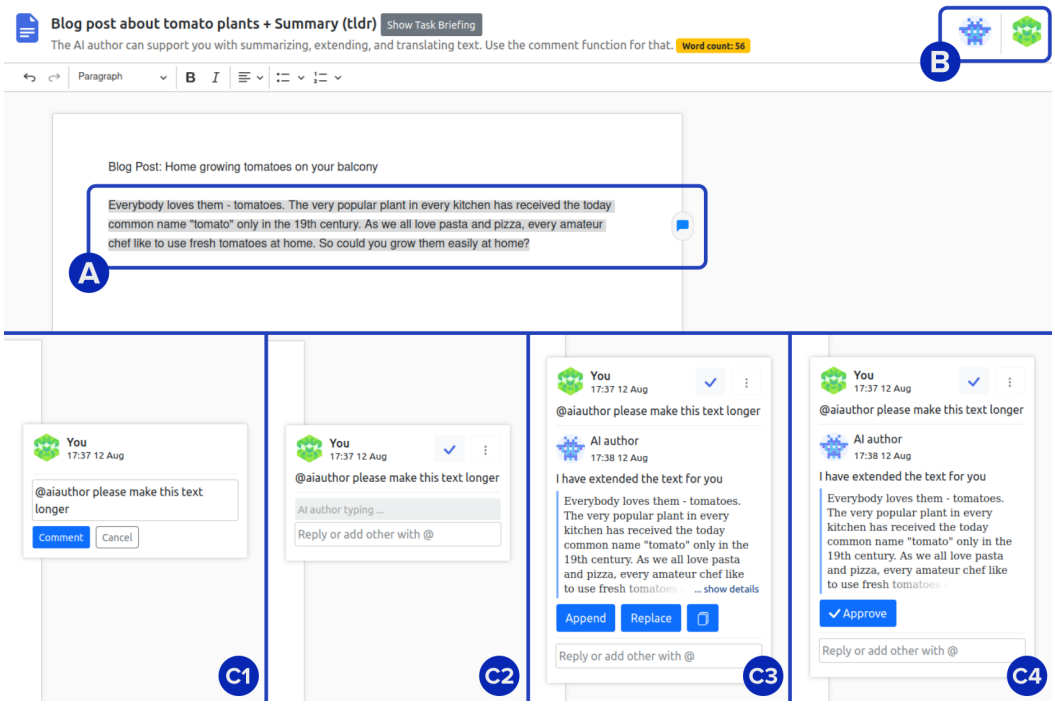


Fig. 5.3 This figure is taken from [P1], it shows screenshots of our first prototype with a conversational UI. The top half of the figure shows the editor. We designed the editor to look similar to existing online text editing tools. Part A shows a user-selected text, with a balloon icon next to it on the right for adding an interactive comment. Part B shows that both the user and the AI are "present" in the document. The bottom half of the figure shows the procedure of delegating a task to the AI via an interactive comment. C1 shows a comment to extend the text. C2 shows an indicator that the AI is carrying out the task ("typing"). In the top right of the comment card, users can approve the comment or use the dropdown to edit or delete the comment. C3 shows the AI's suggestion, three options to accept the text (append, replace, copy to clipboard). Furthermore, users can view more details and potentially write a reply. However, our system did not support multi-turn conversations. C4 shows the comment card after the users have clicked on "append" and the suggestion has been appended in the main text – users can then approve this result, which closes (removes) the comment.

49]. Based on these works, we decided to choose iterative functional prototyping to leverage engineering as a design method. By designing and developing functional prototypes, we actively worked with the possibilities offered by LLMs and explored their integration into workflows. With our functional prototypes, we conducted user studies to collect usage data and observe interaction with the models. This way, we empirically grounded further design decisions for our prototypes. In the following, we describe the design process of our online text editors integrating AI features.

We started the design process by investigating the AI capabilities by researching related work and industry solutions. Motivated by these capabilities, we carefully designed a formative survey to investigate people's writing preferences and how they would delegate different tasks to AI through written language. We combined the findings from our survey

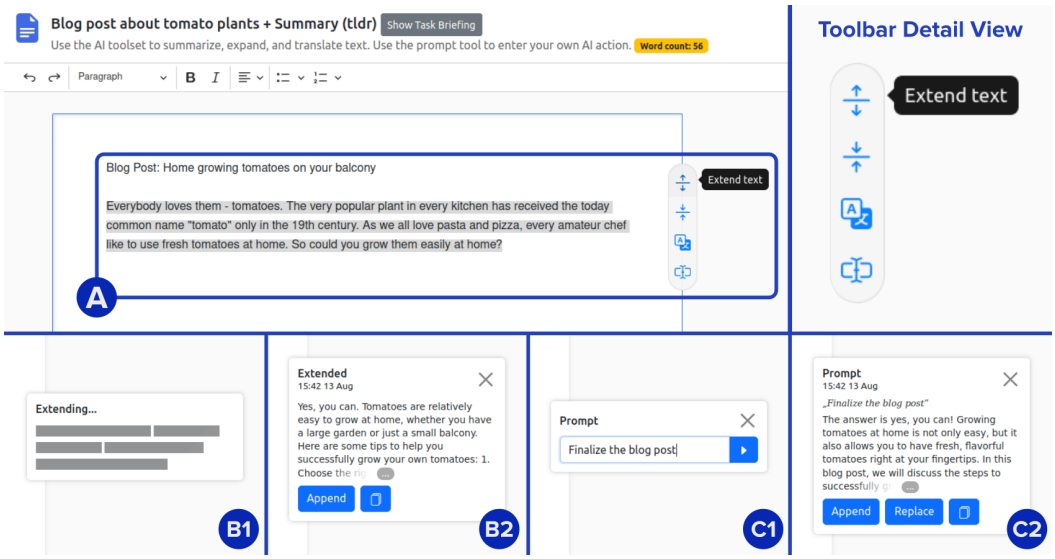


Fig. 5.4 Figure taken from [P1]. Screenshots of our second prototype with an AI toolbar. The top half shows the editor, similar to existing online text editor tools. Part (A) shows how users could select text, which brings up the AI toolbar floating next to it on the right. This toolbar provided buttons for extending, summarizing, translating, and prompting own functions. The bottom half shows the procedure of applying an AI function from the toolbar. (B1) shows a loading indicator after a user has clicked on the extend text function. (B2) shows the AI's suggestion, with buttons for viewing the complete suggestion ("..."), appending it, and copying it to the clipboard. Replacing (cf. Figure 5.3) was not possible for "extend" since it would have overwritten the source text. (C1) shows the UI for prompting own functions. (C2) shows the AI's response to this.

with the results of a design session in our research group. In this design workshop, we gathered sketches on how AI could be integrated into a text editor. The discussion on these sketches resulted in our idea of using the interactive comments feature of an online text editor as a conversational UI for delegating tasks to AI. Inspired by these first sketches, we switched to prototyping the feature with web technologies. By implementing the feature iteratively, we could test and improve the UI and workflow. A robust version of the prototype can be seen in Figure 5.3.

The first prototype allowed users to mark text and delegate a task to an AI agent that was displayed in the UI as also present on the document. This agent carried out the task and presented the output as a reply to the comment. We offered three ways of accepting the AI's output (replace, append, and copy to clipboard). With this prototype, we conducted a first user study. A thorough analysis of the interaction logs and user feedback indicated that repeatedly used AI features should be put into a toolset UI. Based on the results, we iterated on the prototype and replaced the interactive comments (conversational UI) with a toolbar offering specific AI functions and the option for prompting user-defined functions, see Figure 5.4.

With this second version of the prototype, users could mark a text and select predetermined AI functions from a floating toolbar – or alternatively prompt a custom function. AI outputs still appeared as annotations. The outputs were presented in a card element, similar to an interactive comment. Users could view the suggestion in this card and, similar to prototype one, choose from three options to accept the text. We replicated the user study with this prototype. Analyzing the data from the second user study showed that participants combined and chained pre-specified AI functions and switched to prompting for improving existing functions and applying custom functions. Overall, this iterative process of developing and changing the prototype through findings from subsequent user studies shaped our view on how users access the functional space offered by the LLM through the UI.

This demonstration of functional prototyping highlights the importance of empirical data in refining the design of AI functions, offering insights that non-functional prototypes fall short of exposing.

5.4 Evaluating human-AI prototypes on the web

This section is based on the publication

"StudyAlign: A Software System for Conducting Web-Based User Studies with Functional Interactive Prototypes" [P2].

Our related contributions [P1, P3] were based on an iterative design process, in which we switched early to building web prototypes. Through this approach, we gained a clearer understanding of how users interact with generative models while also enabling the measurement of quantitative metrics. Since these prototypes are built for the web, they can be evaluated in web-based user studies [P1]. Related work on human-AI interaction also relied on web prototyping (e.g. [23, 43, 4, 18, 13, 20, 14, 15]). These studies, along with our own experience, demonstrate the applicability of these methods. However, setting up proper experimental procedures, combining prototypes with questionnaires, and integrating data logging can be tedious and complex. To reduce this effort of (repeatedly) setting up web-based field experiments and make research more transparent through sharing features, we developed the software system StudyAlign.

The system builds on four software parts 1) a frontend for participants that guides them through procedures so that online studies are feasible, 2) an admin panel to create and control studies in a team, 3) a TypeScript library that eases data logging in a way that requests to the backend become one liners, 4) a backend server for persisting log data and serving a

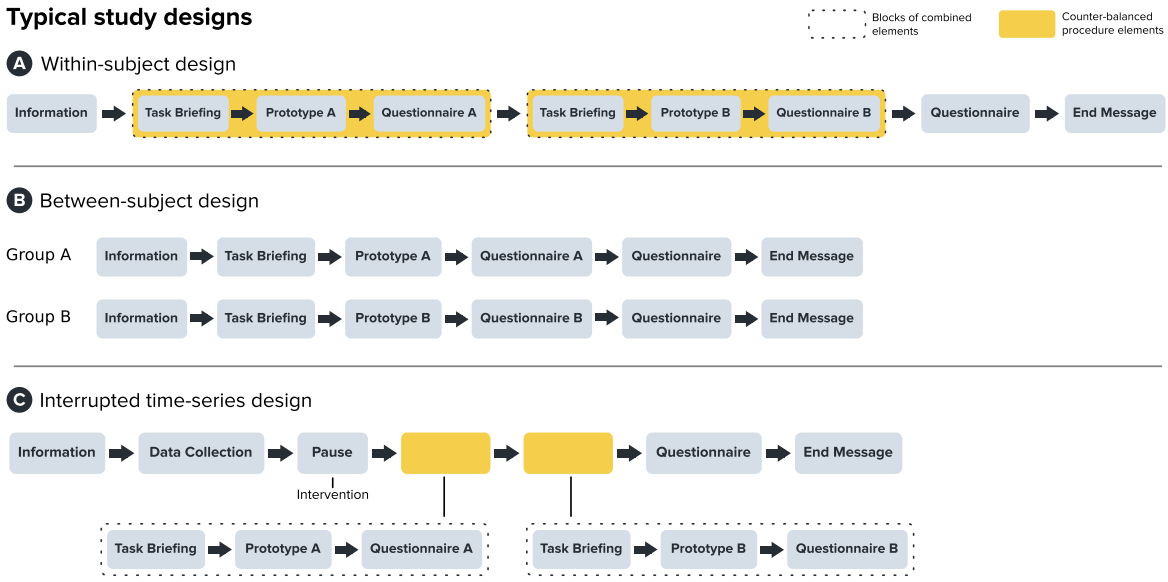


Fig. 5.5 Figure taken from [P2]. This illustration shows typical study designs that can be implemented with StudyAlign. The system supports within-subjects, between-subjects, and interrupted time-series designs. Example A shows a within-subject design: The task briefings, prototypes, and questionnaires in this example are combined into one "block" element. Such blocks of elements are then counterbalanced. Example B shows a between-subjects design: A separate study is created for each group. It is possible to duplicate an existing study, which eases the creation of between-subjects designs. Example C shows an interrupted time-series design: The depicted procedure is similar to a within-subject design but with an initial data collection step and an extended pause (e.g. multiple days), for example, to give researchers the time to create an intervention based on the collected data. This intervention then influences the prototypes in the remaining study procedure.

central API to all parts of the systems. These software parts offer many features to HCI research teams. Some of these features are highlighted in the following. A full list of the features can be found in the original publication [P2]. The features cover the whole cycle of a study: The UI of the admin panel supports managing studies. For example, procedures can be edited with a drag-and-drop editor, and elements can be selected for counter-balancing. While the study is running, the backend allows logging of native browser events, such as mouse, keyboard, and touch events, but it also allows logging of custom data objects. The admin UI offers an overview of currently logged data. After the study is done, the system offers a data export, and study schemes can be shared. This sharing option makes it possible to share procedures easily as a material for papers to support open science.

Implemented in the admin UI, the drag-and-drop editor for setting up procedures allows creating complex study designs. The most important study designs are shown in Figure 5.5. These are actual designs of studies that were conducted with the system by us and other researchers in our network. Through this demonstration across more than ten studies with

StudyAlign, we have proven the feasibility of the system. Some of these collaborations have been published [14, 15, 17, 53, 54, 6].

With StudyAlign, we contribute a systematic methodological foundation for evaluating human-AI interaction via web-based functional prototypes. The system is available as an open-source project [1].

[1] https://studyalign.com

Chapter 6

Functional flexibility in generative AI interfaces

This section is based on the publication

"Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts" [P1].

With our contribution on functional flexibility in generative AI interfaces, we propose a theoretical construct on how user interfaces shape access to the functional space of generative AI, as summarized at the end of this section in Table 6.1. This theoretical construct is a tool for thinking about text-based interaction with generative AI. A key concept of this construct is that LLMs offer an unlimited space of output generations that fulfill conceptual functions from the user's perspective (e.g. inspiration, summaries, drafting). We name this space *the functional space offered by LLMs to users*. UIs for LLMs have two roles for accessing this space. They define *how users can express their intents* (e.g. "I need ideas") and *to which extent they can access the LLM's space of functions*. Both together determine the *degree of functional flexibility* made available to users. A visualization of our theoretical construct can be seen in Figure 6.1.

We derived this construct by synthesizing contributions from related work with our findings and reflections across our empirical studies. Related work by Reynolds and McDonell [35] described interaction with LLMs as "locating an already learned task". We deem this interaction "locating a function". Finding the precise language to locate such functions via prompting in the space of an LLM is termed abstraction matching by Liu et al. [26] and Subramonyam et al. [39] identified verbalizing user-desired goals as the "gulf of envisioning". We consider this user task "formulating intent to locate a function". Our findings showed that

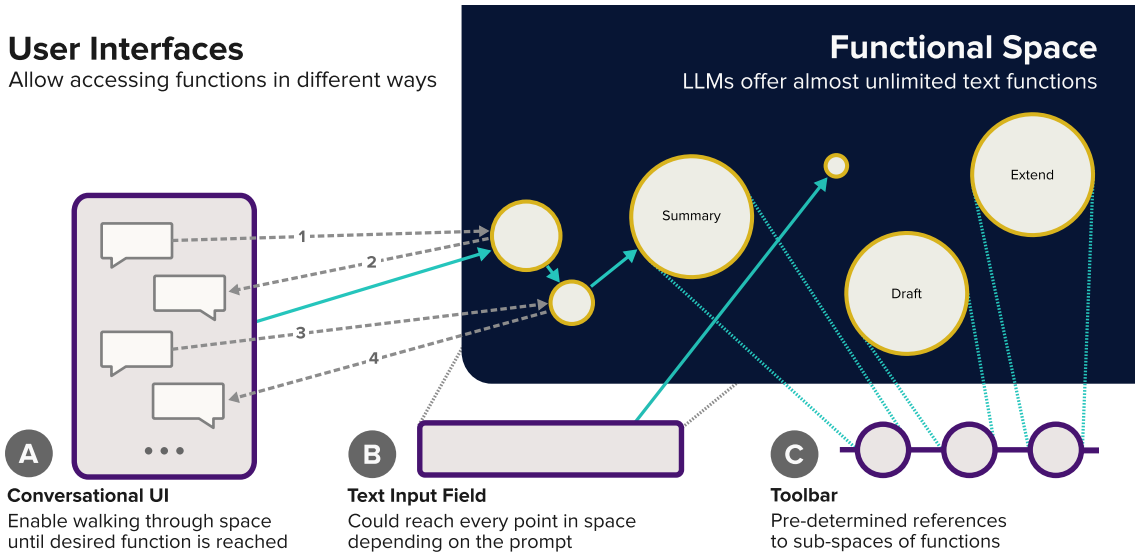


Fig. 6.1 Figure taken from [P1]. The figure shows a high-level illustration of how different degrees of functional flexibility arise from UIs that make the functional space of generative AI accessible to users: The dark plane visualizes the functional space provided by an LLM. To keep this figure generalizable, the space is given no explicit dimensions. Subspaces within this space represent different text functions that users desire to access, i.e. their target functions, such as "summary". Each point represents a specific function (e.g. a specific way to summarize). The illustrated UIs are examples of actual UIs that allow users to dive into the functional space at different locations and limit the reachable subspaces: (A) With a conversational UI, users typically "walk" through the space via messages until a desired function is reached in conversation. (B) With a text input field for free prompting, users could directly reach every point in the space, depending on the prompt. (C) A toolbar offers pre-determined references to subspaces of functions. Overall, these UIs vary in the amount of involved context that is passed to the LLM, the possibility to prompt instructions, and subsequent user interactions on the LLM's responses. Our theoretical perspective emphasizes the role of the UIs as the essential connection between user intent and the functional space of LLMs. UIs define both (1) how users can express their intents and (2) to which degree they can access the LLM's space of functions.

UIs serve a key role of granting users access to the functional space of LLMs: The UI design shapes how users can locate functions, leading to different degrees of functional flexibility (maximum flexibility: free prompting UI; reduced flexibility: specified AI tools). Following work by Beaudouin-Lafon et al. [5], we have taken an analytical, critical, and constructive perspective on our theoretical construct, and compiled a set of key questions that can be asked to practically assess functional flexibility in prototypes (Table 6.2). We apply these questions in Section 6.1 to reflect on our prior work, and in Section 6.2 to analyze related work through the lens of functional flexibility.

The concept of functional flexibility emerged from our four-year-long investigation of writing interaction with AI in multiple studies, as shown in Figure 6.2. In the following, we provide details on these studies and how they support our concept. Our contribution [P1] includes a survey with 121 participants on text editor usage, collaboration, and task delegation

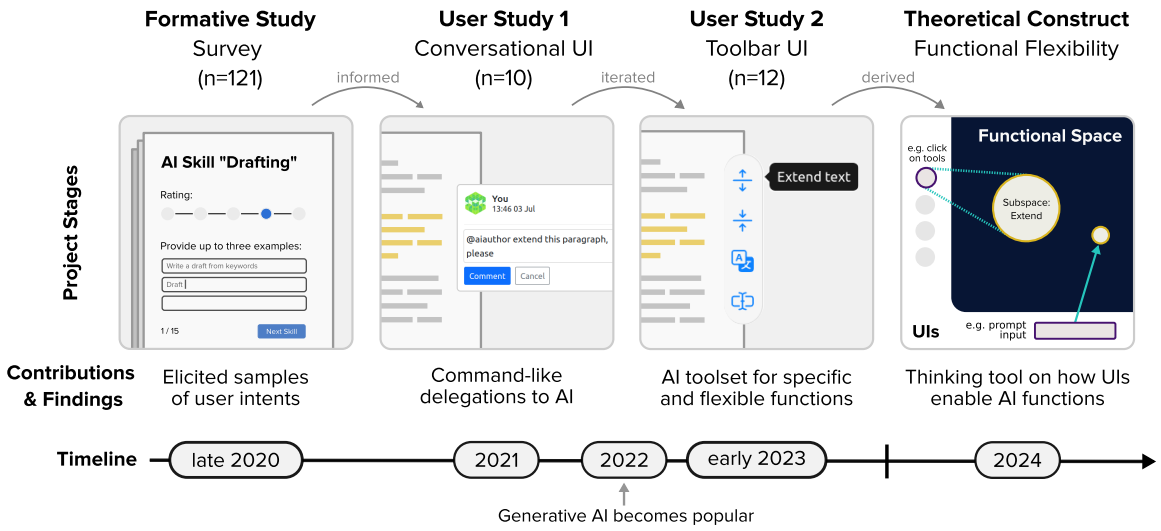


Fig. 6.2 Illustration taken from [P1]. We investigated text editing with generative text models over four years, from 2020 to 2024, covering the moment when generative AI became widely popular. This research effort has four parts (left to right): (1) a formative survey to explore people's text editor usage and to elicit how users would delegate tasks to AI (N=121), (2) a user study with a prototype integrating conversational AI (N=10), (3) an alternative prototype offering AI tools, again tested in a user study (N=12), (4) a theoretical view on UIs and AI with a focus on functional flexibility.

to AI without the "legacy bias" of popular tools such as ChatGPT. A main part of our survey elicited how people at the time would instruct writing tasks to AI (18 AI capabilities in total, e.g. summarizing, extending, translating) through a conversational UI. In terms of functional flexibility, we asked participants for instructions to locate these functions within the space of an LLM. These instructions had a mean length between 4.54 and 4.96 words. Based on these samples, we built an intent recognition component for our first prototype to call different models upon user requests. Furthermore, the survey responses informed the design of our first prototype of a text editor that provided a conversational UI for AI in interactive comments. Users could delegate the writing tasks to summarize, extend, and translate through interactive comments to an AI co-author. We focused on these AI capabilities since they were feasible to implement at that time. In our prototype, the functional space was limited to these three functions, but the UI gave them potentially maximum freedom, similar to free prompting. We conducted a mixed-methods online experiment with 10 participants and found that they used very short, command-like instructions for delegating tasks with a mean length of 1.79 words. The length of instructions was decreased in contrast to the more elaborate prompts elicited in our survey. During the interviews, participants requested pre-determined AI tools, i.e. an AI toolbar. Based on these findings, we revised the prototype, moving the AI into a toolbar and also providing a text input for free-form prompting. This design offered users both specified AI functions and maximum flexibility through prompting. The AI functions

were served by dedicated models, and the free prompting gave users access to an LLM. We replicated our study with the revised prototype. Our second study with 12 participants revealed that AI tools (extend, summarize, translate) accounted in sum for 64.42% of all triggered AI functions. Participants prompted their own functions in 35.57% of all uses of AI. These prompts had a length of 5.59 words. The prompts entered for the free prompting were longer than the prompts in study 1. In light of our theoretical construct of Functional Flexibility, we interpret the user interaction as follows: AI tools, such as buttons in a toolbar, serve as shortcuts to access repeatedly used functions in the functional space of an LLM. Free prompting of functions provides more flexibility to users than delegating a limited set of functions. Yet, the effort increases to specify the intent.

Three concrete UI examples covered across our prototypes and studies serve as an illustration (Figure 6.1) for how they provide a different amount of functional flexibility. We describe the examples as follows: Example A shows a conversational UI. This UI allows for a step-wise navigation of the functional space. The history serves as context, and each utterance pushes through the space until the desired target function is reached. Example B shows a text field for single prompt input. Similar to a conversation, this UI also offers the opportunity to reach every point within the functional space, but with only one try. Free prompting offers a maximum of functional flexibility to the user. Example C shows a toolbar UI. This UI specifies UI functions and constrains the functional flexibility more than the other examples. Toolbar UIs offer pre-determined AI tools to reach commonly desired functions that need to be informed through user research.

Table 6.1 This table is taken from [P1] and provides an overview of the core aspects of our theoretical construct on how user interfaces shape access to the functional space of generative AI.

Aspect	Theoretical approach
Functional Space	LLMs offer great functional flexibility, which offers users an unlimited space of output generations that fulfill desired functions. Thus, an LLM can be considered as a functional space. Users instruct LLMs with text inputs from a similarly unlimited input space to access these functions.
Role of User Interfaces	UIs define how users can express their intents and to which degree they can access the LLM's functional space. UIs allow diving into the functional space at different locations and limit the reachable subspace.
Key Insight	The key mechanism by which "interaction with LLMs" can be construed is accessing a desired target function through suitable UIs.

Overall, our theoretical construct serves as a thinking tool for researchers, designers, and engineers to analyze and reflect on the functional flexibility offered to users. When designing for everyday use, for example, we should consider that we give users maximum flexibility through prompt-based interfaces (e.g. ChatGPT). However, this maximum functional flexibility might not always be in the best interest of the user, as prompting is demanding [52, 41]. Alternatively, toolbar UIs could offer specific AI functions to users, e.g. to access repeatedly used functions, while text input UIs offer access to more flexible AI functions. This example shows that there are many possibilities for interaction with AI going beyond conversing with and prompting LLMs. We imagine that future AI systems integrate closely into users' actual workflows, considering functional flexibility as an explicit design choice.

6.1 Reflecting through the lens of functional flexibility on our previous work

To demonstrate that our theoretical construct of functional flexibility can be applied to other UIs as well, we reflect through its lens on the prototypes from our previous work on mobile writing with AI [P3]. These prototypes enabled users to compose text from phrases (suggestion lists, Figure 5.2 right) and let the AI write text automatically (continuous generated text, Figure 5.2 left). In the following, we describe how functional flexibility varied while interacting with these interfaces and answer analytical questions from Table 6.2. Furthermore, we answer critical and constructive questions from this table by contrasting both prototypes in light of functional flexibility.

Writing with suggestions. We visualized in Figure 6.3 how writing with suggestions enabled accessing the functional space. We go through each step, indicated with the letters A-E: (A) When people write text from suggestion lists, their initial input serves as the defining prompt. The users are free to write this text and have maximum flexibility, and the text input defines the starting area in the functional space of the LLM, i.e. subspace. (B) After that, the system generates a set of phrases that the user can choose from to extend the text. Generating these choices is similar to randomly picking text from the subspace. The user has the control to select from these suggestions. The functional flexibility is determined by the set of suggestions returned by the system. (C) A re-generation of choices triggers another random selection from the initially defined subspace. (D) When the user selects a phrase to extend the text, the area for the next generation of phrases will be adjusted. (E) The user can edit the text anytime to reach maximum flexibility through manual text input, shifting

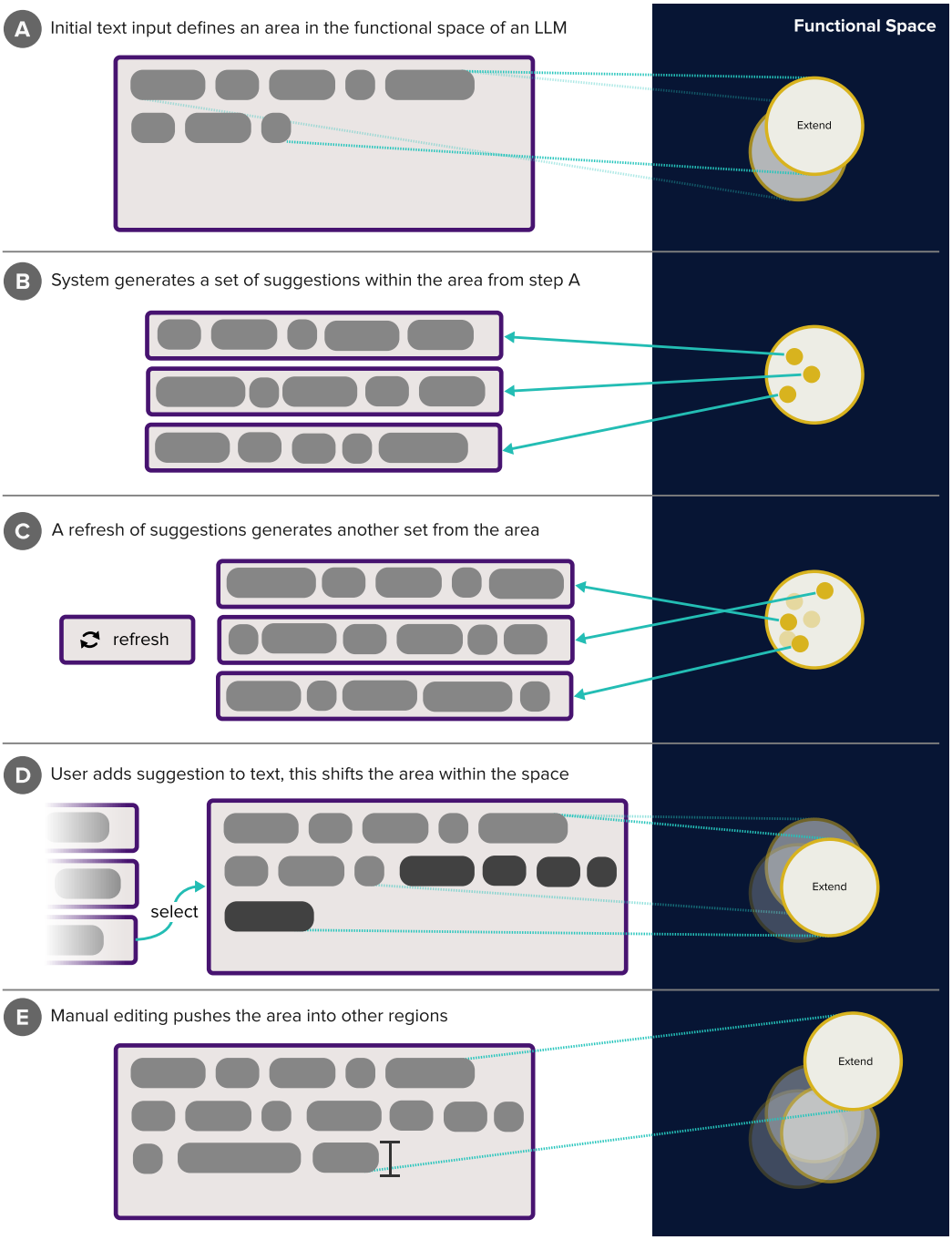


Fig. 6.3 Visualization of how writing with suggestions [P3] enabled accessing the functional space of the model. The figure illustrates how (A) an initial text input defines an area within the functional space, (B) the system generates a set of suggestions within that area, (C) refreshing suggestions generates a new set (D) user selection of a suggestion shifts the area within the functional space, and (E) manual editing pushes the area into other regions.

the selection area in the functional space. Through repeated interaction with the system, the intent will be reached while passing through moments of varying functional flexibility.

Continuous generated Text. More drastically is the difference of functional flexibility with our second example, where the system writes continuously. We visualized in Figure 6.4 how writing with AI continuously generating text enables accessing the functional space: (A) The user writes the initial text in a free-text input field. This input defines the initial prompt and grants maximum flexibility to the user. This prompt defines the starting area in the functional space, i.e. the subspace whose function represents extensions of the original text. (B) From there, the system writes the text word by word. It picks an extended version of the text from the subspace, and the UI animates the writing word by word. Each call of an extended version expands or shifts the subspace. The user has no flexibility at this point. (C) The user can delete text to undo the generation and trigger a new generation from the system. There is a chance that the system will explore another direction when it starts writing again. Deleting text and initiating a new generation leaves the user with minimal functional flexibility. (D) However, the user can edit the text manually, which offers increased flexibility again. That way, the user has the chance to capture another part of the space, and let the system write automatically from the new location again. This example shows a drastic switch between maximum flexibility and no flexibility.

Table 6.2 Table taken from [P1]: It contains a set of questions that can be asked to interpret the analytical, critical, and constructive power of functional flexibility implemented in a system [5].

Perspective	**Questions**
Analytical	What UI elements are available with which degree of functional flexibility? What mix of functional flexibility is offered by the UI / interaction design overall?
Critical	How does the degree of functional flexibility influence the interaction with the system? How does the offered degree of functional flexibility improve the system? Does the system/UI offer the right degree of functional flexibility to the user for reaching/identifying their goals? What problems does the degree of functional flexibility offered by the system cause? Can users reach/identify their goals with the degree of functional flexibility offered by the system?
Constructive	Can functional flexibility inspire new ideas when designing a new system? (Inspired by [28]) How can users be supported to reach their goals? How can we help users identify their goal? Which UI elements could be added to a design to increase/decrease functional flexibility?

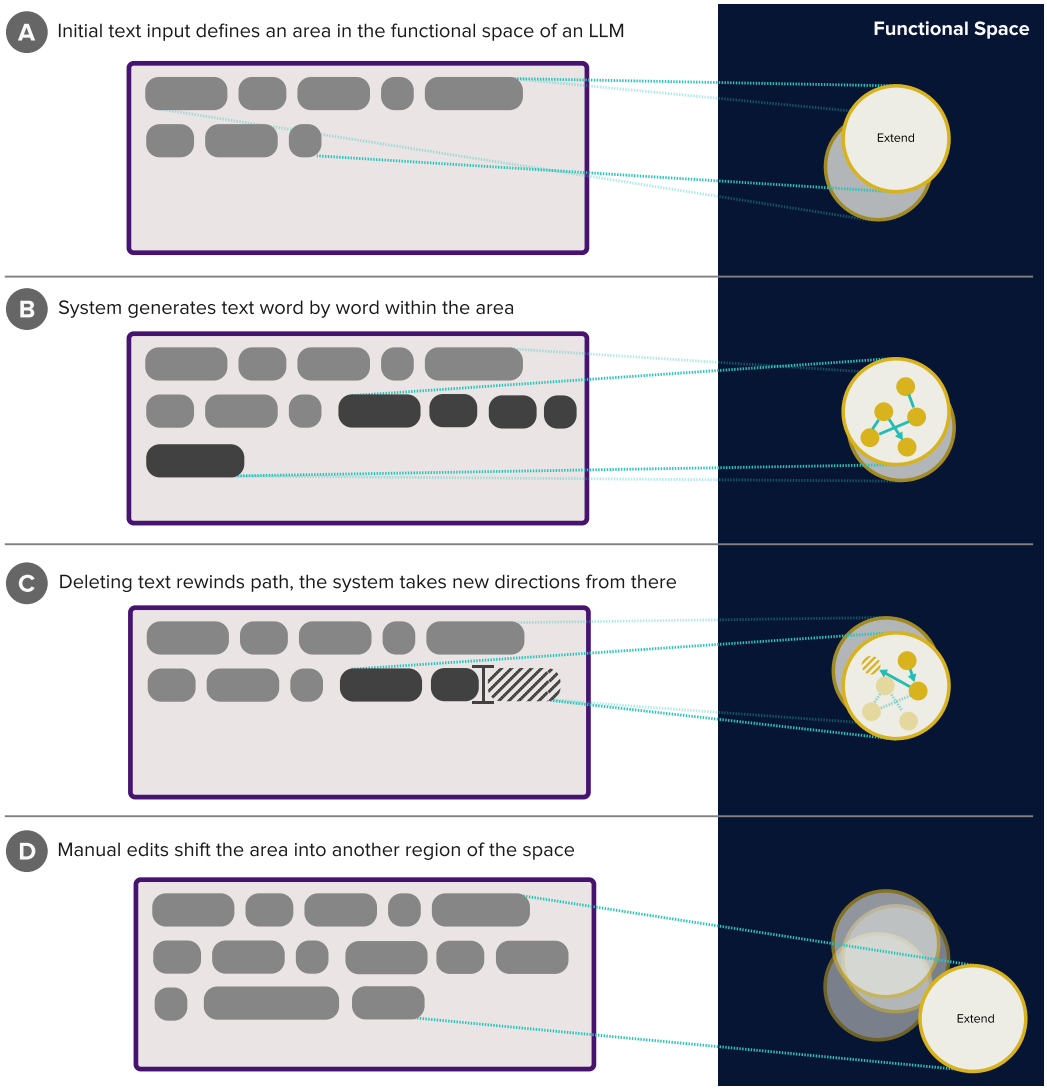


Fig. 6.4 Illustration of how writing with continuously generated text [P3] enabled accessing the functional space of the model. The process begins with (A) an initial text input defining a specific area within the functional space. (B) The system continuously generates text, exploring possibilities within that area, expanding or shifting the area by adding text. (C) Deleting text rewinds the path, allowing the system to take new directions from the revised point. (D) Manual edits shift the area into other regions of the functional space.

Contrasting both systems in light of functional flexibility. Both prototypes allow dynamic switches of flexibility. However, the difference is in the reduced amount of functional flexibility. Writing with suggestions preserves a confined level of flexibility, while a system that writes text continuously offers this flexibility only for manual deletions and edits. Critically, this difference influences the interaction with the system: Even with a low functional flexibility, users can actively explore the functional space, for example, by selecting suggestions or drawing a new set of suggestions from the space. In contrast, when the system writes the text word by word, the user observes how the system explores the space independently. The users only edit the output to reach their goals, resulting in increased text deletions, as our data showed. Constructively, UI elements could be added to alter the functional flexibility for systems that continuously generate text, for example, adding controls to "rewind" and "playback" generations to steer the exploration of the space in the process of writing the text word by word. Retrospectively, we demonstrated that our concept allows discussing the functional flexibility of systems as we have introduced them in [P3]. In general, such discussions of functional flexibility allow for contextualizing findings and could lead to new hypotheses, for example, that the amount of functional flexibility influences the perceived authorship when writing with AI.

6.2 Functional flexibility in related work

We further demonstrate that Functional Flexibility can also be applied to UIs from related work. As an example, we discuss the interface from the application Script&Shift by Siddiqui et al. [36] through the lens of our construct by answering analytical, critical, and constructive questions from Table 6.2.

In Figure 6.5, we visualized how writing with the tool Script&Shift enabled accessing the functional space of an LLM. We focus on their features "text elaboration" and "merging layers": (A) When people type a forward slash "/", an inline toolbar appears, offering six "Writer's Friends", such as Idea Ivy for ideation, and Detail Danny for elaborations. These Writer's Friends refer to subspaces of functions in the functional space. Users can select one of these Friends to scope a subspace. The tools confine the functional space of the LLM for the user. (B) An input field for free prompting appears below the paragraph, and gives the user maximum flexibility to navigate the subspace referenced through the previously selected Writer's Friend. The user is free to specify further instructions to the model, for example "expand this text without adding any additional details" (cf. [36]). (C) By conditioning the LLM's generation on the user's entered prompt, the system generates two suggestions. The user can choose one of the two suggestions for including it in the original text. (D) A

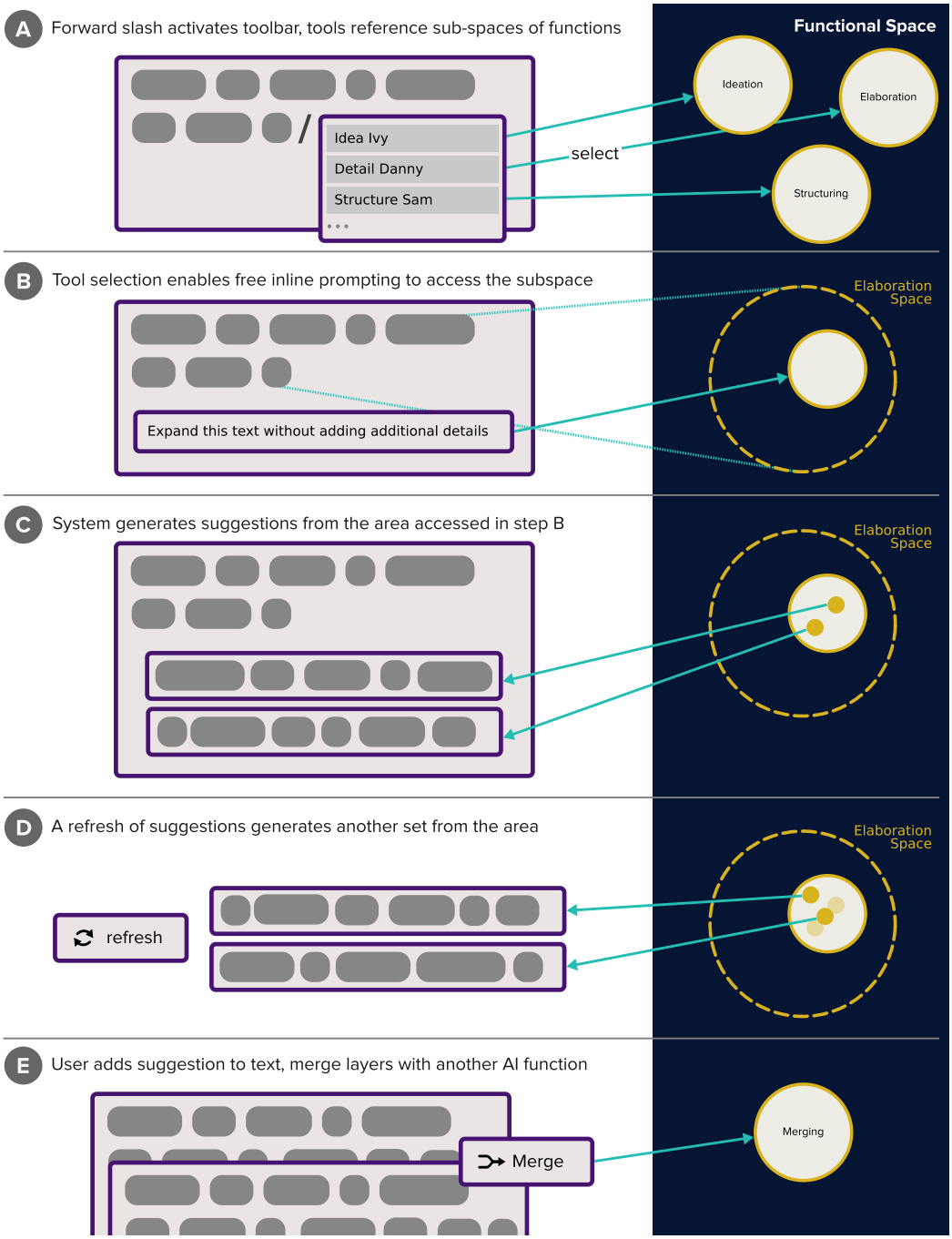


Fig. 6.5 Illustration of how Script&Shift [36] enabled accessing the functional space of the LLM. The illustration shows how (A) typing a forward slash triggers displaying an AI toolset. Users can select one of these functions. (B) Selecting a tool scopes a subspace, and an inline free prompting text field allows users to explore this subspace. (C) The system generates a pair of suggestions. (D) Refreshing suggestions generates a new set. (E) Users can add the suggestion to the text. If multiple layers exist, they can be combined through a pre-determined merge function.

re-generation of the suggestions triggers the system to serve two other suggestions from the same subspace, as in the previous step. (E) With Scrip&Shift, it is possible to write in multiple layers. Their concept of layers is similar to unconnected pages that can be interacted with on an infinite canvas space. Some Writer's Friends generate such pages as an output. For example, Structure Sam generates two differently organized variants of the current page. Such pages can be merged into a single page by letting the LLM combine them.

Dissecting the UI and the interaction flow in this way shows that the application involves a combination of an AI toolbar and an inline text input field for free prompting, similarly illustrated in our Figure 6.1 in example B. Selecting a tool (Writer's Friend) from the AI toolset (list of Friends) scopes a subspace (confined flexibility) and the free prompting UI element gives back maximum flexibility to navigate this subspace. Dragging and dropping one page onto another, scopes again a pre-determined subspace (confined flexibility) to let the LLM merge the two pages. This answers the analytical question from Table 6.2 "What UI elements are available with which degree of functional flexibility?". The AI toolbar offers six pre-determined functions, which guide the user with decreased functional flexibility. However, some users might want to specify their own functions for maximum flexibility, i.e. free prompting without selecting one of the predefined Writer's Friends before. These users could benefit from defining their own Writer's Friend to reach their goals, and thus more freely explore the functional space. This answers the critical question "Can users reach/identify their goals with the degree of functional flexibility offered by the system?". Adding another trigger, such as typing a square "#" could open an inline text input field for free-prompting own functions. These functions could be named and added to the toolbar by checking a box next to the input field and adding a name. This way, users could add repeatedly used AI functions as further Writer's Friends. This free-prompting of functions (maximum flexibility) and making them available as reusable Writer's Friends (confined flexibility) would allow for dynamic switches of functional flexibility. This answers the constructive question "Which UI elements could be added to a design to increase/decrease functional flexibility?".

Reflecting on Script&Shift through the lens of Functional Flexibility enabled us to dissect the interaction analytically, as well as constructively criticize the application. This demonstrates the value of Functional Flexibility as a theoretical thinking tool for building human-AI interaction.

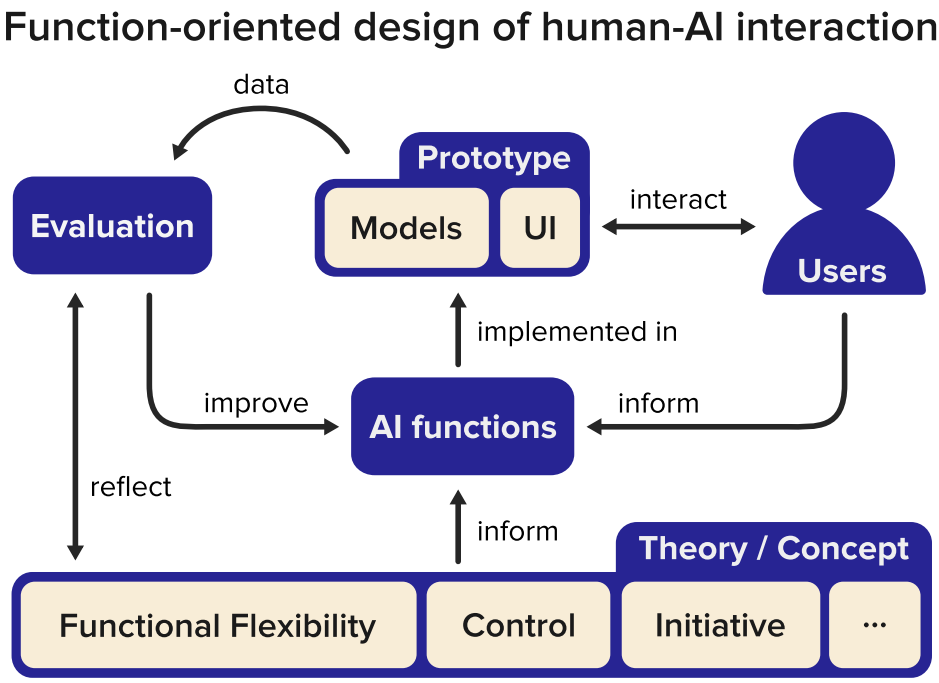


Fig. 6.6 The iterative cycle of *function-oriented design for human-AI interaction*. The illustration depicts how theory and concepts (such as functional flexibility, control, and initiative) and users inform the AI functions implemented in prototypes. Users interact with prototypes, generating data that feeds into evaluation. Insights from evaluation are then reflected upon to improve the AI functions as implemented through models and UI, creating a continuous loop of refinement and enhancement in human-AI interaction design.

6.3 Function-oriented design of human-AI interaction

The final outcome of this thesis is a vision of function-oriented design of human-AI interaction that goes beyond prompting – interaction that is designed carefully for AI functions to serve user intents. Together, the contributing publications exemplify how to approach such function-oriented designs. Thus, this section will connect our publications to advocate for integrating generative models into UIs following a function-oriented design approach. We have visualized the process of function-oriented design of human-AI interaction in Figure 6.6.

THEORY and CONCEPT: Theories and concepts inform AI functions from a theoretical perspective. They serve as a grounding for evaluation but can also be reflected on in the evaluation step (bottom half of Figure 6.6). As our work on functional flexibility has shown [P1], it is important to offer the right amount of functional flexibility to users for enabling them to achieve their goals effectively. Our theoretical construct serves as a foundation for design decisions on human-AI applications. It can be used as a thinking tool to answer questions such as "How much freedom should users have to explore the functional space?"

and “How should users express their intents?”. Comprehending UI solutions with our thinking tool is also important since different levels of functional flexibility built into UIs influence user interaction with AI.

INTERACTIVE PROTOTYPES: Prototypes implement AI functions that are informed by theory and concepts initially and then refined iteratively (top center of Figure 6.6). Building functional interactive prototypes is central to the function-oriented design of human-AI interaction. Experiences from our own work and related work have shown that formative surveys and early user studies can be used to refine such prototypes (e.g. [P3, P1, 14, 15, 6, 53, 54]). With functional interactive prototypes, it is possible to explore actual interaction early in the design process [P1]. Such functional prototypes allow logging interaction data and collecting user feedback for evaluating the design of AI functions. Modified versions of the prototypes can be tested in subsequent user studies.

ITERATE THROUGH EVALUATION: Our process involves iterating prototypes through their functions based on evaluating user data (top half of Figure 6.6). Collecting feedback and data early in the design process provides evidence to base design decisions on. Our software system StudyAlign [P2] supports the evaluation of human-AI prototypes since it allows logging data from users, running subsequent studies, and scaling studies. Findings from an evaluation can be used to reflect on theories and concepts, as we have done in our work [P3], where we suggested considering initiative more concretely as a design material. Furthermore, the evaluation findings support the improvement of AI functions, such as moving to different implementations of AI functions [P1].

UNDERSTAND USERS: Users inform the AI functions from a practical perspective (c.f. functional flexibility funnel [P1]). Users interact with the prototype and provide qualitative and quantitative data (top right of Figure 6.6). Concretely, people make use of intelligent features (AI functions) in certain ways to reach personal goals. As our work on mobile typing [P4] has shown, users select word suggestions for different functions than only increasing performance, such as completing or correcting a word, but also to select and modify it afterwards. Similar, in our user studies on functional flexibility [P1], people delegated tasks by following their own strategies for using the AI features. These examples illustrate that the actual use of human-AI features often diverges from the initial assumptions (e.g. by system builders). Such insights can inform iterative improvements of functions, fostering human-centered interaction with AI.

FINAL OUTCOME: In combination, these four aspects highlight important building blocks for integrating AI functions into human-AI systems. Following these aspects substantiates designing, implementing, and evaluating human-AI interaction – arriving at functional human-AI interaction and offering the right amount of flexibility to users.

Chapter 7

Discussion

7.1 Interaction beyond prompting interfaces

In our research on functional flexibility [P1], we proposed to think about the role of UIs for accessing the space of functions offered by LLMs. We highlight that the design of UIs defines how flexible users can access conceptual functions within this functional space (e.g. drafting text, gaining inspiration). UI elements vary the degree of functional flexibility. For example, pre-determined tools such as buttons in a toolbar offer a low flexibility, while a text field for free prompting offers a high flexibility.

It might be tempting to offer maximum flexibility to users, since a single input field for free prompting replaces any other UI elements. We can see this strategy in products such as ChatGPT or Microsoft Copilot. These products are marketed as a promise of maximum flexibility offered through a text input box. However, our conceptual view on UIs and LLMs shows that prompting interfaces shift the responsibility for locating AI functions to the user. We think prompting is an important technique to instruct models and will not disappear, but the effort to identify functions should not be moved from designers and engineers to users. Instead, designers and engineers could identify and implement pre-determined AI functions that are often requested by users (e.g. through user studies). This way, designers and engineers could build features that fulfill certain user needs: a UI for an "inspire" feature would look different from a "summarize" feature, for example, providing different controls for adjusting the input and visualizations for presenting the output. Thus, we argue it is important to understand which AI functions users are looking for and how UIs enable them to access such functions in the wide space offered by LLMs. Our theoretical contribution on functional flexibility supports researchers, designers, and engineers in this process to imagine and reflect on UIs for AI.

7.2 Switching early to functional prototyping and iterative improvements

A central method of our research is building functional prototypes and improving them iteratively for creating novel interaction and understanding user behavior. At the time of starting this thesis, HCI research identified challenges using machine learning as a material for UX design [49], and particular challenges in sketching for natural language interfaces [50]. They suggested abstracting language and modeling capabilities as a way to sketch for technology offering natural language interfaces. Different from their approach, our approach is focused on prototyping. We sketch only first ideas and switch to prototyping early in the design process. Instead of following a design process thoroughly, such as the user-centered design process, we take shortcuts and design through engineering the prototypes. This approach allows us to learn from actual interactions and not only from abstractions of interaction. Our findings have shown that people prompted differently than expected from our formative study [P1] – the interaction diverged from the concept. Such insights allowed us to iterate the feature from a conversational UI to a tool-based UI. To ease the evaluation of this engineering-driven design process, we have created StudyAlign [P2], a system for conducting web-based experiments and online studies with functional interactive prototypes. In summary, we show that research on human-AI interaction benefits from functional prototyping. By contributing our tools as open-source projects [1], we hope to motivate other researchers to choose the same approach.

7.3 Initiative as a design parameter of user roles in human-AI interaction

An important aspect of this thesis was exploring the concept of initiative in human-AI interaction. Interfaces that allow the system to take an active role allow for mixed-initiative interaction [21]. In our work on writing with suggestion lists and AI writing continuously [P3], we suggested to consider initiative a variable "design material" for designing mixed-initiative interfaces. With this thesis, we take a more detailed view on initiative in human-AI interaction. We define initiative for generative models as follows: "models can take the initiative to do something", and argue based on our contribution on functional flexibility [P1] that initiative has facets. In the context of text editing, for example, the AI system could take initiative in many ways, such as editing a piece of text, but also leaving a comment

[1] https://github.com/studyalign

for feedback and inspiration. Concretely, these different types of initiative are reflected in user roles, as suggested in [P3]. Such roles could be a co-author: someone who edits a text (initiative to edit the text), and an advisor: someone who gives feedback on how to improve the text (initiative to provide feedback). These findings show that initiative has facets and is reflected in user roles. As part of reflecting on this thesis, we thus recommend, in practice, emphasizing AI user roles in the design process, as they share inherent types of initiative.

7.4 Limitations & Reflections

This section is taken and adapted from our work on functional flexibility [P1] as it captures the limitations and reflections of this thesis as well.

7.4.1 Methodological and technical limitations

In our work [P1, P3], we followed a mixed-methods approach and observed participants' screens via video calls. This approach offered insights we otherwise would have missed. People also shared their thoughts *during* writing, which not only provides a glimpse at their thinking processes but also helps us to interpret the findings from interaction logging (cf. [8]).

Overall, this investment comes with typical tradeoffs of qualitative research, such as favoring detailed observation over a larger sample and risking that observations might influence people's behavior.

We do not claim to have achieved an "optimal" implementation of the AI features, and this would be a moving target anyway, since model development advances quickly. "Perfect" text generation was not required for our research here, as long as AI-generated text could be realistically included in writing workflows. This bar was met for the tasks in our studies, based on our own assessments and people's perceptions.

7.4.2 Reflections on research in a fast-moving field

Working on this thesis, across the widespread appearance of generative AI, revealed many opportunities for reflection. These aspects present a more personal reflection on research practice and community, which we deem important to share for such a long-running project as well.

Time pressure Generative models have become widely accessible to the public and a ubiquitous topic in HCI research during my PhD. The releases of new generative models

and tools drastically increased the pace for research on human-AI interaction. For many in HCI, this puts pressure on implementing prototypes and user studies, in particular, to claim novelty. For instance, we anecdotally noted increased sharing of preprints of papers that later appeared at HCI conferences with a double-anonymous review process. Some actively advertised their work on social media already during review phases. Thus, time pressure around a trending topic affects our HCI research culture, with possibly negative consequences on anonymity in reviewing.

Competition Contrasting the paper proceedings and workshops of CHI 2022, 2023, and 2024 reveals that human-AI research and generative models have explosively grown into a ubiquitous topic in HCI. Subjectively, conference discussions on ongoing work, including our own, seemed more careful than before. We think that early-career researchers, in particular, feel a pressure of competition that is exacerbated for "hyped" topics. While competition, to some extent, is positive and a part of research work, as a community, we should actively explore how we can avoid that such pressure becomes an obstacle to fruitfully exchanging ideas in our community and hearing the voices of junior scholars.

Dependencies on technical developments While conducting a user study, it might happen that new models and capabilities become available as products that put months of work into question. Moreover, widespread use of new products rapidly changes people's legacy bias. For example, the release of ChatGPT shifted prior experiences in our second study in [P1]: 75% reported experience with ChatGPT, i.e. a specific design of conversational AI. In many ways, our survey and first study had anticipated conversations with LLMs for writing, giving us the perhaps unique opportunity of analyzing user expectations on delegating writing tasks to an LLM without influence by prior use of ChatGPT. At the same time, industry developments enabled us to easily integrate free-text prompting in our second study. In general, to navigate such shifts, research on human-AI interaction could interpret generative models as a material for design, similar to Yang et al. [49], but also as a material for prototyping that is useful in researching fundamental concepts, such as control, initiative, agency, and ownership. Interpreting generative models as a material conceptualizes them as an exchangeable element and thus might reduce the dependency on specific models. This, in turn, might support HCI researchers in extracting generalizable insights.

7.5 Future work

Research is an ongoing endeavor. Thus, I give an outlook for future research directions based on this thesis.

Running field studies: We have created an online text editor with AI features for our studies [P1]. We plan to make the editor publicly available and log interactions with our StudyAlign logging backend in a longitudinal field study to learn about realistic interaction with LLMs and gain external validity of findings. Currently, there is few research in this direction [27] and we think the research community could benefit from such insights.

Multiagent and multiuser interaction: We have built an agent into our first version of the text editor prototype [P1]. This agent was experimental, and its functionality was limited. With recent developments such as software frameworks like AutoGen [46] it becomes feasible to build and research multiagent interaction. Another opportunity for future research is exploring multiuser setups with AI. Current research on human-AI interaction investigates only single-user interaction with AI. To the best of our knowledge, there is no work on multiuser interaction. To fill this gap, we have conducted a user study with an extended version of our original text editor [P1] that is now supporting multiple agents and multiple users to allow for true collaboration. In the user study [25], we found that teams incorporated AI agents into existing norms of authorship, control, and coordination, rather than treating them as equal team members. The work is currently under review.

Chapter 8

Conclusion

I conclude this thesis with a final reflection on the timeline of this research and prompting as an interaction technique for LLMs. Finally, I summarize my contributions and recommend moving towards function-oriented design of human-AI interaction.

At the beginning of my PhD in 2020, it was unclear how to design for human-AI interaction in applications [49, 50]. Two years later, conversational AI appeared and was made available to the public. This event brought prompting to almost every human-AI interaction in practice. However, related work has shown that prompting is challenging [52, 41]. As argued by Morris [30], prompting interfaces might not be optimal for end users. Our research covered the key moments in the development of AI features such as ChatGPT and also shows that prompting is oftentimes hard and does not sufficiently reflect the users' needs [P1]. Instead, users can benefit from pre-determined AI tools for specific tasks. Certainly, prompting will not disappear since it is a viable method for controlling LLMs, but end-users could benefit from different additional interfaces and methods to fulfill their needs. In particular, our theoretical construct on Functional Flexibility enables actively thinking (analyzing, criticizing, and constructing) about the differences between designs of human-AI interfaces to envision such interfaces. We anticipate that future interaction techniques and UIs for human-AI interaction will go beyond prompting interfaces. Indeed, we have recently seen research in this direction [29, 54].

In this dissertation, we explored Functional Flexibility over the course of four years and contributed comprehensive insights into how user interfaces allow users to express their needs and access the almost unlimited possibilities offered by text-based generative models. To inform functions, we conducted user surveys and derived examples of AI functions from related work and industry solutions. To analyze functions in an established UI, we examined the user motivations for involving word suggestions in typing on mobile keyboards and identified different strategies for choosing word suggestions that go beyond efficiency, show-

ing that even seemingly straightforward, established UIs with AI are used to serve various functional needs. Experimentally, we investigated how fundamental interaction concepts such as control and initiative influence writing with AI functions and found that designing AI for these concepts alters how users perceive themselves as authors. To implement and iterate functions, we chose functional prototyping as a core method for our experiments. We built such functional prototypes early in the research process to collect realistic usage data and feedback from users to iterate on our designs of AI functions, again implemented in revised prototypes. For example, we prototyped interaction for mobile writing with AI and online text editors for writing with conversational AI, AI toolbars, and prompts. To evaluate the functions in our prototypes, we built StudyAlign, a software system for conducting web-based experiments with functional prototypes, serving as a methodological foundation for our studies.

Overall, this thesis demonstrates how to integrate LLMs into human-AI systems and examines how users apply models in writing workflows. Our findings inform the design, implementation, and evaluation of human-AI interaction: Concretely, we recommend exploring UIs for human-AI interaction that go beyond prompting. To facilitate this, our theoretical contribution on Functional Flexibility offers a foundation to construct, analyze, and comprehend user interfaces for human-AI interaction. In conclusion, we propose the vision of "function-oriented design of human-AI interaction", a research-informed design approach for creating human-AI interaction that goes beyond prompting interfaces and focuses on user goals.

References

[1] Saleema Amershi et al. “Guidelines for Human-AI Interaction”. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI ’19*. ACM Press, 2019, pp. 1–13. DOI: 10.1145/3290605.3300233.

[2] Ian Arawjo et al. “ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–18. DOI: 10.1145/3613904.3642016.

[3] Kenneth C. Arnold, Krysta Chauncey, and Krzysztof Z. Gajos. “Predictive text encourages predictable writing”. In: *Proceedings of the 25th International Conference on Intelligent User Interfaces*. IUI ’20. Association for Computing Machinery, 2020, pp. 128–138. DOI: 10.1145/3377325.3377523.

[4] Kenneth C. Arnold, Krysta Chauncey, and Krzysztof Z. Gajos. “Sentiment Bias in Predictive Text Recommendations Results in Biased Writing”. In: *Proceedings of the 44th Graphics Interface Conference*. GI ’18. Canadian Human-Computer Communications Society, 2018, pp. 42–49. DOI: 10.20380/GI2018.07.

[5] Michel Beaudouin-Lafon, Susanne Bødker, and Wendy E. Mackay. “Generative Theories of Interaction”. In: *ACM Transactions on Computer-Human Interaction* 28.6 (2021), pp. 1–54. DOI: 10.1145/3468505.

[6] Karim Benharrak et al. “Writer-Defined AI Personas for On-Demand Feedback Generation”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–18. DOI: 10.1145/3613904.3642406.

[7] Dan Bennett et al. How does HCI Understand Human Autonomy and Agency? arXiv:2301.12490 [cs]. 2023. DOI: 10.1145/3544548.3580651.

[8] Advait Bhat et al. “Interacting with Next-Phrase Suggestions: How Suggestion Systems Aid and Influence the Cognitive Processes of Writing”. In: *Proceedings of the 28th International Conference on Intelligent User Interfaces*. ACM, 2023, pp. 436–452. DOI: 10.1145/3581641.3584060.

[9] Sid Black et al. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. Version 1.0. 2021. DOI: 10.5281/zenodo.5297715.

[10] Tom B. Brown et al. “Language Models are Few-Shot Learners”. In: *arXiv:2005.14165 [cs]* (2020). arXiv: 2005.14165.

[11] Daniel Buschek, Martin Zürn, and Malin Eiband. “The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers”. In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, 2021, pp. 1–13. DOI: 10.1145/3411764.3445372.

[12] Aaron Chatterji et al. How People Use ChatGPT. Tech. rep. w34255. National Bureau of Economic Research, 2025, w34255. DOI: 10.3386/w34255.

[13] John Joon Young Chung et al. "TaleBrush: Sketching Stories with Generative Pre-trained Language Models". In: *CHI Conference on Human Factors in Computing Systems*. ACM, 2022, pp. 1–19. DOI: 10.1145/3491102.3501819.

[14] Hai Dang et al. "Beyond Text Generation: Supporting Writers with Continuous Automatic Text Summaries". In: *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. ACM, 2022, pp. 1–13. DOI: 10.1145/3526113.3545672.

[15] Hai Dang et al. "Choice Over Control: How Users Write with Large Language Models using Diegetic and Non-Diegetic Prompting". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, 2023, pp. 1–17. DOI: 10.1145/3544548.3580969.

[16] Graham Dove et al. "UX Design Innovation: Challenges for Working with Machine Learning as a Design Material". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2017, pp. 278–288. DOI: 10.1145/3025453.3025739.

[17] Fiona Draxler et al. "The AI Ghostwriter Effect: When Users Do Not Perceive Ownership of AI-Generated Text But Self-Declare as Authors". In: *ACM Transactions on Computer-Human Interaction* (2023), p. 3637875. DOI: 10.1145/3637875.

[18] Katy Ilonka Gero, Vivian Liu, and Lydia Chilton. "Sparks: Inspiration for Science Writing using Language Models". In: *Designing Interactive Systems Conference*. ACM, 2022, pp. 1002–1019. DOI: 10.1145/3532106.3533533.

[19] Matthew Guzdial and Mark Riedl. "An Interaction Framework for Studying Co-Creative AI". In: *arXiv:1903.09709 [cs]* (2019). arXiv: 1903.09709.

[20] Han L Han et al. "Textlets: Supporting Constraints and Consistency in Text Documents". In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems - CHI '20*. ACM, 2020, p. 13. DOI: 10.0.4.121/3313831.3376804.

[21] Eric Horvitz. "Principles of mixed-initiative user interfaces". In: *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. CHI '99. ACM Press, 1999, pp. 159–166. DOI: 10.1145/302979.303030.

[22] Anna Kantosalo and Hannu Toivonen. "Modes for Creative Human-Computer Collaboration: Alternating and Task-Divided Co-Creativity". In: *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*. Sony CSL, 2016, pp. 77–84.

[23] Mina Lee, Percy Liang, and Qian Yang. "CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities". In: *CHI Conference on Human Factors in Computing Systems*. ACM, 2022, pp. 1–19. DOI: 10.1145/3491102.3502030.

[24] Florian Lehmann. "Mixed-Initiative Interaction with Computational Generative Systems". In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. CHI EA '23. Association for Computing Machinery, 2023. DOI: 10.1145/3544549.3577061.

[25] Florian Lehmann, Krystsina Shauchenka, and Daniel Buschek. Collaborative Document Editing with Multiple Users and AI Agents. 2025. arXiv: 2509.11826 [cs.HC].

[26] Michael Xieyang Liu et al. ""What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, 2023, pp. 1–31. DOI: 10.1145/3544548.3580817.

[27] Tao Long, Katy Ilonka Gero, and Lydia B Chilton. "Not Just Novelty: A Longitudinal Study on Utility and Customization of an AI Workflow". In: *Designing Interactive Systems Conference*. ACM, 2024, pp. 782–803. DOI: 10.1145/3643834.3661587.

[28] Wendy E. Mackay and Michel Beaudouin-Lafon. "Interaction Substrates: Combining Power and Simplicity in Interactive Systems". In: *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. ACM, 2025, pp. 1–16. DOI: 10.1145/3706598.3714006.

[29] Damien Masson, Young-Ho Kim, and Fanny Chevalier. Textoshop: Interactions Inspired by Drawing Software to Facilitate Text Editing. arXiv:2409.17088 [cs]. 2025. DOI: 10.1145/3706598.3713862.

[30] Meredith Ringel Morris. "Prompting Considered Harmful". In: *Communications of the ACM* 67.12 (2024), pp. 28–30. DOI: 10.1145/3673861.

[31] Santiago Negrete-Yankelevich and Nora Morales-Zaragoza. "The apprentice framework: planning and assessing creativity". In: *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2014)*. computationalcreativity.net, 2014, pp. 280–283.

[32] Kseniia Palin et al. "How do People Type on Mobile Devices?: Observations from a Study with 37,000 Volunteers". In: *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 2019, pp. 1–12. DOI: 10.1145/3338286.3340120.

[33] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: (2019), p. 24.

[34] Colin Raffel et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs, stat]. 2023.

[35] Laria Reynolds and Kyle McDonell. "Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm". In: *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, 2021, pp. 1–7. DOI: 10.1145/3411763.3451760.

[36] Momin Siddiqui, Roy Pea, and Hari Subramonyam. Script&Shift: A Layered Interface Paradigm for Integrating Content Development and Rhetorical Strategy with LLM Writing Assistants. arXiv:2502.10638 [cs]. 2025. DOI: 10.48550/arXiv.2502.10638.

[37] Nikhil Singh et al. "Where to Hide a Stolen Elephant: Leaps in Creative Writing with Multimodal Machine Intelligence". In: *ACM Transactions on Computer-Human Interaction* (2022), p. 3511599. DOI: 10.1145/3511599.

[38] Hendrik Strobelt et al. "Interactive and Visual Prompt Engineering for Ad-hoc Task Adaptation With Large Language Models". In: *IEEE Transactions on Visualization and Computer Graphics* (2022), pp. 1–11. DOI: 10.1109/TVCG.2022.3209479.

[39] Hari Subramonyam et al. “Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–19. DOI: 10.1145/3613904.3642754.

[40] Sangho Suh et al. “Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–26. DOI: 10.1145/3613904.3642400.

[41] Lev Tankelevitch et al. “The Metacognitive Demands and Opportunities of Generative AI”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–24. DOI: 10.1145/3613904.3642902.

[42] Jörg Tiedemann and Santhosh Thottingal. “OPUS-MT – Building open translation services for the World”. In: *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*. Ed. by André Martins et al. European Association for Machine Translation, 2020, pp. 479–480.

[43] Kashyap Todi et al. “Conversations with GUIs”. In: *Designing Interactive Systems Conference 2021*. ACM, 2021, pp. 1447–1457. DOI: 10.1145/3461778.3462124.

[44] Ashish Vaswani et al. “Attention is all you need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Curran Associates Inc., 2017, pp. 6000–6010.

[45] Justin D. Weisz et al. “Design Principles for Generative AI Applications”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 2024, pp. 1–22. DOI: 10.1145/3613904.3642466.

[46] Qingyun Wu et al. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. arXiv:2308.08155 [cs]. 2023. DOI: 10.48550/arXiv.2308.08155.

[47] Tongshuang Wu, Michael Terry, and Carrie J. Cai. “AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts”. In: *arXiv:2110.01691 [cs]* (2021). arXiv: 2110.01691.

[48] Qian Yang et al. “Investigating How Experienced UX Designers Effectively Work with Machine Learning”. In: *Proceedings of the 2018 Designing Interactive Systems Conference*. DIS ’18. Association for Computing Machinery, 2018, pp. 585–596. DOI: 10.1145/3196709.3196730.

[49] Qian Yang et al. “Re-examining Whether, Why, and How Human-AI Interaction Is Uniquely Difficult to Design”. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI ’20. ACM Press, 2020, pp. 1–13. DOI: 10.1145/3313831.3376301.

[50] Qian Yang et al. “Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence”. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI ’19*. ACM Press, 2019, pp. 1–12. DOI: 10.1145/3290605.3300415.

[51] Ann Yuan et al. “Wordcraft: Story Writing With Large Language Models”. In: *27th International Conference on Intelligent User Interfaces*. ACM, 2022, pp. 841–852. DOI: 10.1145/3490099.3511105.

[52] J.D. Zamfirescu-Pereira et al. “Why Johnny Can’t Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts”. In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, 2023, pp. 1–21. DOI: 10.1145/3544548.3581388.

[53] Tim Zindulka et al. Content-Driven Local Response: Supporting Sentence-Level and Message-Level Mobile Email Replies With and Without AI. arXiv:2502.06430 [cs]. 2025. DOI: 10.1145/3706598.3713890.

[54] Tim Zindulka et al. Exploring Mobile Touch Interaction with Large Language Models. arXiv:2502.07629 [cs]. 2025. DOI: 10.1145/3706598.3713554.

Appendix: Original Contributing Publications

Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts

FLORIAN LEHMANN, University of Bayreuth, Germany
DANIEL BUSCHEK, University of Bayreuth, Germany

Many recent user interfaces (UIs) promise great flexibility via access to Large Language Models (LLMs) through prompting. This shifts efforts in accessing functionality from developers to users. However, is a maximum of functional flexibility always beneficial? We find: no – by applying functional flexibility as a lens on our long-term exploration of integrating LLMs into a text editor. In a prompt-elicitation survey, people (N=121) envisioned elaborate prompts for delegating flexible tasks to AI. In contrast, when interacting with a prototype (N=10), users preferred short command-like prompts, as prompting repeated functionality is tedious. Another prototype offered such commands in a toolbar, in addition to flexible prompting, leading users (N=12) to dynamically work with both. We discuss functional flexibility as a theoretical construct with analytical, critical, and constructive value. Overall, we encourage designing UIs for generative AI that go beyond free text prompting by explicitly considering functional flexibility in design.

CCS Concepts: • **Human-centered computing** → **Empirical studies in HCI**; **Text input**; • **Computing methodologies** → **Natural language processing**.

Additional Key Words and Phrases: human-AI interaction, conversational AI, AI tools, writing, large language models, LLM, text editing, functional flexibility, functional space

ACM Reference Format:
Florian Lehmann and Daniel Buschek. 2018. Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 49 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Prompting is in focus in current HCI research (e.g. [2, 29, 53, 54]) but also in industry. Microsoft, for example, has integrated chatbots based on large language models (LLMs) into various tools to extend their functionality (e.g. see Figure 4 in [56]). At the same time, research showed that prompting is demanding [56, 68], and Morris [39] recently argued that such prompting interfaces might not be optimal for end users.

We challenge this current focus on prompting with the terminology of *functional flexibility* that we introduce in detail in this paper: With prompts, users express their intents that serve underlying goals. For example, a prompt for gaining inspiration might say "Give me inspiration for an essay about recent advances in technology". Conceptually, the user's intended function of this prompt is to "inspire" the user. LLMs, such as generative pre-trained transformers (GPTs) [12, 44] are *flexible* in executing such intent-driven functions, since they posses an almost unlimited number of capabilities for generating or transforming text. That is, they offer a broad *functional space*. Thus, the distinctive value of these models for users is their *functional flexibility*.

In this view, the role of user interfaces (UIs) is to make this functional flexibility available to end users. Currently, entering prompts as text is the predominant UI choice for this (e.g. ChatGPT). This grants users a *maximum* of functional

Authors' Contact Information: Florian Lehmann, florian.lehmann@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany; Daniel Buschek, daniel.buschek@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany.

Manuscript submitted to ACM

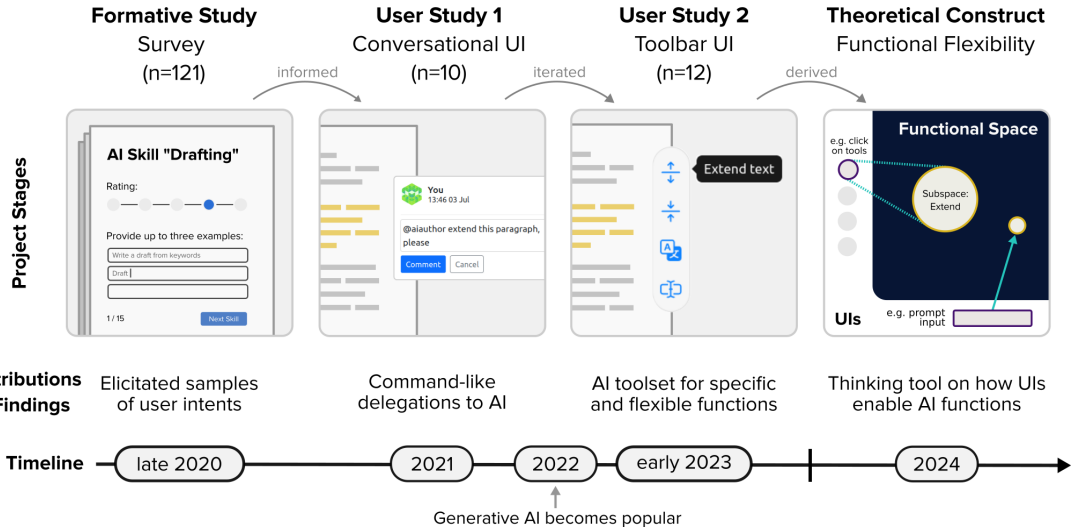


Fig. 1. We investigated text editing with generative text models over four years from 2020 to 2024, covering the moment when generative AI became widely popular. This research effort has four parts (left to right): (1) a formative survey to explore people's text editor usage and to elicit how users would delegate tasks to AI (N=121), (2) a user study with a prototype integrating conversational AI (N=10), (3) an alternative prototype offering AI tools, again tested in a user study (N=12), (4) a theoretical view on UIs and AI with a focus on functional flexibility. In summary, we contribute a set of elicited samples of user intents, findings from our studies about AI use in text editors, and a theoretical perspective on how UIs shape functional flexibility.

flexibility, because free text input enables them to specify *any* request and access a broad range of model outputs (with some degree of quality). However, maximizing functional flexibility might not always be in the best interest of the users, for example, when it comes at the expense of knowing how to achieve something (cf. [56]) – e.g. an empty chat box gives little guidance on what the system can do.

We thus argue that it is important for HCI researchers and practitioners to understand which level of functional flexibility is given to users through various UI designs, to be able to make this an *explicit* design choice. Toolbar UIs, for example, could offer specific AI functions to users as buttons, while text input UIs offer access to more flexible AI functions. A UI might also provide both. In this way, applications could (also) offer a set of pre-determined AI functions to the user without demanding a user prompt. This example of AI tools shows that there are more possibilities for interaction with AI than conversing with and prompting models. Thus, when designing UIs for AI, we see the need to better understand how AI can be integrated into users' actual workflows, considering functional flexibility.

As our main contribution, we propose and develop this concept of *functional flexibility* in the context of interaction with LLMs. We arrive at this through reflection on our multi-faceted research efforts on text editing with LLMs, spanning 2020 to 2024. Concretely, we developed the concept through insights into user expectations and prompting behaviour assessed before the widespread use of prompting LLMs (i.e. release of ChatGPT), followed by further empirical work on prompting behaviour in writing with AI after that moment (see Figure 1). This long-term perspective allowed us to see contrasts and variations in user behaviour and model capabilities, revealing the role of UIs as shaping functional flexibility and users' exploration of the functional spaces provided by LLMs. In more detail, (1) our formative survey (N=121) elicited how users envision delegating writing tasks to AI. This informed our design of a conversational UI (CUI) prototype for text editing with AI. (2) A mixed-methods user study (N=10) revealed that people often entered

very short instructions. (3) Our second prototype then provided these directly as shortcuts in a toolbar UI, in addition to prompting, evaluated in a second study (N=12).

We found that – unbiased by exposure to current LLM-based chatbots – users optimized for command-like instructions in actual interactions and did not negotiate about model output via conversations, although people had envisioned longer prompts in the survey. This changed when adding a toolbar for common functions: The resulting mix of toolbar and free text prompting was used to dynamically switch between pre-specified AI functions and investing in longer prompts for flexible functions. Moreover, across both studies, we found that CUIs allow for parallel workflows with AI. Finally, with the toolbar, users integrated AI functions closely into text editing for specific tasks and chained functions to reach their goals. With *functional flexibility*, we brought together these insights in a new theoretical construct as a thinking tool for researchers and designers. Inspired by Generative Theories of Interaction [6], we compiled a set of analytical, critical, and constructive questions to reflect on our theoretical construct. Finally, we deconstruct and discuss a recent example of a feature-rich human-AI interface from related work [49] to demonstrate the applicability of our theory.

Overall, our multi-faceted exploration of human-AI interaction encourages researchers, designers, and developers of interactive AI systems to move beyond today's predominantly conversational UIs for generative AI. Concretely, we recommend doing so by analyzing and reflecting explicitly on how different UIs shape users' access to the functional space of generative AI models.

The paper is structured as follows: First, we motivate functional flexibility as a fundamental concept in light of the literature (Section 2). Second, we report on our empirical work: a prompt-elicitation survey (Section 3), and two subsequent user studies with prototypes, which focused on a conversational UI (Section 5) and its evolution (Section 6), leading to a version with toolbar plus prompting (Section 7). We then synthesize our findings with contributions from related work and describe in detail our theoretical construct of functional space and functional flexibility (Section 8). Throughout this and the following discussion (Section 9), we consider functional flexibility from analytical, critical, and constructive perspectives [6], along with additional findings.

We release our prototype and material in this project repository to support future research: https://osf.io/qh46n/?view_only=e8b7ab117cea48118d192e5eac7048bf

2 Related Work

Many recent intelligent writing tools use LLMs based on transformer models [60]. Task-specific models serve one purpose such as translating [58] and summarizing text [45]. General purpose models [12, 16, 34, 44], such as GPTs, generalize well across language tasks, leveraging huge training corpora. Before such models were available, research on writing support systems introduced tools surrounding the writing process, for example, to enhance consistency [22], facilitate planning [35], and support creativity [37]. With such models, many intelligent writing tools now generate suggestions and involve prompting [29]. For implementing our prototypes, we combined task-specific models for summarizing and translating text, as well as a general-purpose models for generating text.

2.1 Writing with generated suggestions

Several studies examined the impact of writing with AI-generated text suggestions: For example, integrating sentence-level suggestions in emailing is more demanding than message-level suggestions yet users perceive a higher sense of agency [20]. Presenting multiple suggestions instead of only one is valued for inspiration [13]. Writing with suggestions on mobile devices resulted in longer text but higher perceived authorship than editing AI-generated text [31]. Writing

with text suggestions can influence users' views [24] and lead to more predictable text [3]. Suggestions can be shown constantly [13], after inactivity [8] or on user request [17, 30]. Overall, studies such as these show that writing with phrase suggestions influences how users write and perceive text. Bhat et al. [8] extended the cognitive process model of writing accordingly.

Our prototypes let participants delegate writing tasks to AI, and they could then decide whether to include the results into their draft, in three ways (replace, append, copy) – and further edit it in all cases.

2.2 Writing with prompts

Prompting is a common interaction technique for delegating tasks to LLMs. Related work has found positive effects, for example, that prompting during writing is used to gain inspiration and explore topics [15], and to create own creative writing tools [55]. However, prompting comes with challenges [68], including debugging and evaluating effectiveness [25], and added meta-cognitive demands [56]. Despite these challenges, when prompting is integrated into text editors, it is typically considered a tool for creatives [29]: For example, Yuan et al. [67] built an editor with generative capabilities accessed via prompting in a sidebar and conducted a study with novelists. Their tool increased writers' engagement, and users found it more helpful compared to only using AI text continuations.

Similar to their work, we created an editor with generative capabilities. We restricted AI to a low level of initiative and let participants use different UIs (conversational UI, toolbar UI, prompting) to control text generation. Next, we give an overview of the fundamental concepts of initiative and control for designing human-AI interaction, and suggest that functional flexibility is one such concept.

2.3 Fundamental concepts in designing for human-AI interaction

Two fundamental concepts in designing for human-AI interaction are initiative and control. In mixed-initiative applications [23], AI can take actions on its own. For example, related work investigated agent-initiated interactions with voice assistants [46], initiative as a design parameter [14], and evaluated a collaborative moodboard tool with different levels of initiative and control [28]. While rarely explicitly defined in these papers, initiative is typically understood as starting and carrying out actions, while control concerns adjusting parameters and constraining system functionalities.

Control is an important aspect in interaction with generative AI, such as for text generation [29]. The sense of control is an often discussed aspect in HCI research [7]. Technically, such control can be facilitated by fine-tuning models for specific task [69] or with prompting techniques, such as chaining prompts [65]. Related, Singh et al. [51] reflect on the users' desire to steer the semantics of what is generated by LLMs. This underlines the importance of designing human-AI interaction in a way that addresses user desires, as users want to reach a goal by integrating LLM-based features into their workflows (i.e. reaching a user-expected function).

We suggest that the flexibility of such functionality is another fundamental concept in designing for human-AI interaction. Concretely, to design effective interfaces, we need to understand which functions users expect from generative applications and focus on designing UIs for accessing these functions in the LLM's space of possible functions. In this paper, we investigate this as a fundamental concept across multiple studies, with an emphasis on the role of the UI.

2.4 Summary

In summary, the functional flexibility of interactive generative models is currently overlooked as a distinct concept, as is evident, for example, from the absence of such a concept in comprehensive and widely referenced design guidelines

for interaction with AI [1, 62]. Today, UI and interaction design for LLM-based systems often provide a maximum of the model's functional space to the user through free prompting. This maximizes functional flexibility but also shifts the task of identifying and determining functions from designers and developers to the end-user. This can be challenging for users, since giving effective instructions to an AI model is often difficult and cognitively demanding [56, 68]. In conclusion, we diagnose that, amidst the promises of prompting, there currently is no explicit consideration of the provided functional flexibility when designing interactive LLM-based systems. This motivates our investigation and perspective in this paper.

3 Survey: Eliciting how users would delegate tasks to AI

We conducted a formative survey in late 2020 to elicit how people would delegate writing tasks to an AI for text editing. In regard of functional flexibility, the survey focused on how users express their writing intent to access a certain AI capability. We were originally motivated to collect samples to build an intent recognition system for conversational AI in this context. Now, this data additionally provides rare insight into how people "prompted before knowing about prompting" from systems like ChatGPT. Furthermore, we designed the survey to understand which tools people prefer for editing text documents and how they communicate and collaborate with others. Overall, this combination of elicited text samples and understanding the context in collaborative workflows provided the foundation for the design and implementation of our first prototype. We focus on reporting the analysis of task delegation phrases, and provide the results on tool usage, communication, and collaboration behavior, and the complete questionnaires in our project repository: https://osf.io/qh46n/?view_only=e8b7ab117cea48118d192e5eac7048bf.

3.1 Survey design

We designed the survey to cover the collaborative writing workflow as comprehensively as possible. We researched examples from industry and academia to ask about realistic, typical tasks that people might be interested in delegating to an AI for text editing (overview in Table 1).

3.2 Survey structure

We structured the survey into six sections: workflow, collaboration, writing stages, coordination, AI capabilities, and demographics. Brief descriptions of these sections can be found in Table 2.

3.3 Participants

We consider data from 121 participants (62 female, 54 male, 4 preferred not to say, 1 other). In total, we recruited 122 participants via Prolific[1] but removed the data of one participant due to a mismatch with the pre-screening requirements (language: English). People's age ranged from 18 to 70 years, with a median of 32 years. Self-rating of English language proficiency revealed that 101 were native speakers, 15 were proficient, 1 had advanced knowledge, and 4 had intermediate knowledge. All participants were compensated with £ 8.8 per hour.

3.4 Coding of survey responses on delegating tasks to AI

As a central part of our survey, we asked participants to provide phrases they would like to tell an AI agent in the context of a text document editor, for the writing stages *outlining*, *drafting*, *revising*, and *coordinating*. Moreover, we

[1] https://prolific.co

Table 1. An overview of potential tasks users could delegate to AI. These tasks correspond to specific AI capabilities and were derived from examples from research and industry. Our survey elicited written examples for delegating these tasks by asking participants to write down how they would instruct an AI assistant in a text editor environment to perform them.

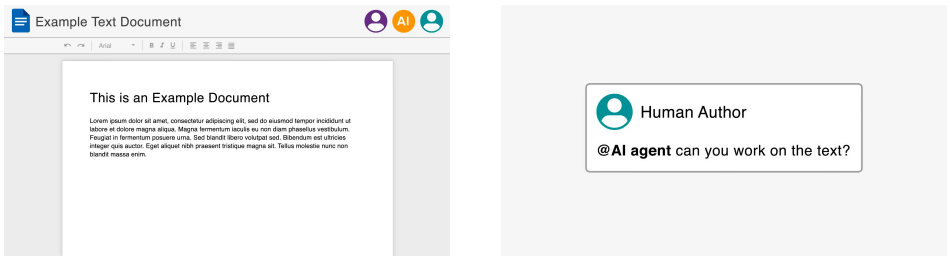
AI Capabilities (Tasks)	**Description**	**Found in**
Revise	Revise a section of text (i.e. change or correct the text).	[57]
Summarize	Summarize a section of text.	[21, 57]
Format	Change a text's formatting.	[57]
Shorten	Shorten a section of text.	[19, 21]
Extend, Elaborate	Extend/Elaborate on a sentence or paragraph that is currently rather short.	Industry: [43]
Highlight	Highlight a part of the text.	[57]
Cut	Cut a section of text that is redundant or unnecessary.	[57]
Generate: Complete sentence	Complete an already started sentence.	Industry: [43]
Generate: Draft paragraph from keywords	Generate a complete paragraph from an inline list of keywords.	[14, 66]
Generate: Inspire/suggest	Generate a draft for a given topic.	[14]
Find and add fact and/or reference	Find fact and add reference.	[18, 66]
Find non-textual material (e.g. photos, illustrations)	Find images or graphics that fit to the theme of the text.	[27, 28]
Translate	Translate a paragraph to the given language.	[44]
Question answering (e.g. domain question)	Answer a question by the user in the context of the text.	[44]
Send	Send a review request to another user.	[9, 57]
Ideation	Make suggestions on an outline.	[35]
Create outline/structure/writing plan	Suggest a text outline.	[35, 66]
Collaboration	AI facilitates collaboration between human writers (e.g. resolve conflicts, highlight changes, notify others)	[18]

elicited phrases to give certain tasks to the AI, such as revising or extending a text. In total, we collected phrases for 18 AI capabilities in this way (Table 1).

To understand the elicited phrases in detail, we coded them for various categories, such as goals, types, and implied AI roles. We inductively coded all phrases with a team of four coders and followed a three-step process. We first proposed initial categories and codes, and discussed them with the goal of finding about eight to twelve categories that could potentially cover the responses across all questions. This led to an initial codebook, which presented an overview of all categories, with descriptions and a list of codes. Furthermore, it contained concrete example decisions/reasonings for each category. Second, each coder independently coded the first 25 rows of each category based on the initial codebook. Coders were allowed to suggest new codes for existing categories while coding. We extensively discussed these first codings in multiple meetings to align our coding decisions and find consensus across coders. Third, we coded

Table 2. The structure of our survey.

Section	Topic & Description
1	**Workflow with text documents** General writing experience, tool preferences
2	**Collaborative workflow** Communication preferences, tools for collaboration
3	**Writing stages (Outlining, Drafting, Revising)** Asking for up to five phrases that participants would like to be able to tell an AI assistant to support them during the respective writing stage.
4	**Coordinating work on a text** Asking for up to five phrases that participants would like to be able to tell an AI assistant to support them with coordinating work on a text (questions, task assignments, etc.).
5	**AI capabilities** Rating of usefulness. Asking to suggest phrasings for giving a task to AI. Up to three example sentences for each AI capability, see Table 1.
6	**Demographics** Demographics, such as age and occupation



(a) Editor Mockup (b) Comment Mockup

Fig. 2. Two mockup screenshots we presented to survey participants to demonstrate how a text editor could look like that allows for delegating tasks to AI. (a) Displays the AI appearing on the document like a user (top right icons), while (b) displays a UI element with a written task delegation to AI. We intentionally decoupled the visual representation of the task delegation from the editor to put focus on the process of delegating the task, not on a specific UI design.

all samples. Each coder coded the responses for four to five AI capabilities and for one writing stage. Afterwards, our team performed a round-robin check of each others' codings. Finally, the lead coder streamlined all codings and flagged low-quality samples.

3.5 Coding results

We coded a total of 1447 phrases that participants would use with AI in the specific writing stages and to coordinate working on a text, as well as 3206 phrases for the specific AI capabilities. In this section, we focus on reporting on the coded categories "Style", "Goals / Motivation", and "Co-Creative Role". Furthermore, we focus our report on the AI capabilities *extending*, *translating*, and *summarizing*, since we deemed those technically feasible for our prototype at the time. Tables with details about these codings can be found in Section A.

Table 3. Overview of our coding of elicited task delegations in the four writing stages: outlining, drafting, revising, and coordinating. Around 90% of elicited task delegations for AI were valid (we removed samples that were flagged as low quality by the coders). The average text length was approximately 6 words per task delegation. Reading ease was computed as the mean Flesch Reading Ease Score: Delegations scored "fairly easy" in the coordinating stage and "plain English" in the others.

Writing Stage	Samples	Removed	Valid	Text Length (words)	Reading Ease
Outlining	444	67	377 (84.91%)	M=5.59, SD=3.14, min=1, max=21	64.84
Drafting	407	50	357 (87.71%)	M=5.30, SD=3.31, min=1, max=19	62.66
Revising	403	41	362 (89.83%)	M=5.32, SD=3.17, min=1, max=18	62.71
Coordinating	390	39	351 (90.00%)	M=6.96, SD=3.77, min=1, max=22	71.58

Table 4. Overview of our codings of elicited task delegation inputs for using specific AI capabilities. We covered 18 AI capabilities but focused on the three capabilities in bold, namely summarize, extend, and translate. 83.60% to 95.59% of inputs were valid, e.g. we removed samples with a low quality. The average text length was between four and five words. The mean Flesch Reading Ease Score showed delegations to AI for summarizing a text as "difficult to read", and delegations to AI for extending and translating as "plain English".

AI Capability	Samples	Removed	Valid	Text Length (words)	Reading Ease
Revise	204	12	192 (94.12%)	M=4.14, SD=3.10, min=1, max=18	62.60
Summarize	**189**	**31**	**158 (83.60%)**	**M=4.96, SD=6.28, min=1, max=54**	**56.61**
Format	223	14	209 (93.72%)	M=4.32, SD=2.70, min=1, max=16	61.20
Shorten	206	33	173 (83.98%)	M=4.68, SD=3.95, min=1, max=40	64.38
Extend	**194**	**26**	**168 (86.60%)**	**M=4.54, SD=2.91, min=1, max=18**	**66.55**
Highlight	198	12	186 (93.94%)	M=4.41, SD=4.41, min=1, max=54	64.24
Cut	206	14	192 (93.20%)	M=4.32, SD=2.81, min=1, max=18	65.70
Complete	191	27	164 (85.86%)	M=4.52, SD=3.32, min=1, max=22	55.59
Draft	198	25	173 (87.37%)	M=5.49, SD=3.28, min=1, max=19	71.77
Inspire	193	19	174 (90.16%)	M=5.50, SD=4.24, min=1, max=34	64.61
Add Reference	205	12	193 (94.15%)	M=4.56, SD=3.13, min=1, max=17	49.30
Find Non-Text Material	216	34	182 (84.26%)	M=5.03, SD=2.68, min=1, max=17	65.72
Translate	**204**	**9**	**195 (95.59%)**	**M=4.67, SD=3.20, min=1, max=21**	**62.92**
Question Answering	210	23	187 (89.05%)	M=5.03, SD=2.83, min=1, max=17	77.73
Send	206	7	199 (96.60%)	M=4.93, SD=3.12, min=1, max=18	80.67
Ideation	185	43	142 (76.76%)	M=4.96, SD=2.93, min=1, max=14	68.29
Create Outline	197	33	164 (83.25%)	M=4.68, SD=2.72, min=1, max=15	62.09
Collaboration	204	49	155 (75.98%)	M=5.86, SD=3.30, min=1, max=19	56.47

3.5.1 Eliciting phrases based on writing stages. Table 3 shows descriptive statistics. The length of the phrases varied between 1 and 22 words. The mean length per stage was between 5.32 and 5.59 words; only Coordinating had 6.96. The coded "goals" (Table 12) showed that participants aimed for inspiration and learning in the outlining stage. For example, participants wrote phrases such as "AI agent please suggest a document structure" and "look up this heading in wikipedia". In the drafting and revising stage, they focused on efficiency and improving text, such as "Check content for

punctuation", "Check for flow and ease of reading" when drafting, and "Correct spelling errors" and "Reword this text for me" when revising. The predominant goal in the Coordinating stage is coordination, and less focused on efficiency, for example, "Divide tasks evenly amongst contributors" and "Keep track of to-do tasks".

As shown in Table 13, the coding of input "styles" revealed that imperative inputs (e.g. "Suggest additions") appeared the most, while almost a quarter of inputs were questions (e.g. "Can you give some feedback on this?"). Regarding "roles" (Table 14), participants looked primarily for AI as an assistant, in particular for coordinating, for example, with phrases like "Send the document out for review or approval". For outlining, they also referred to the AI in a co-author role, with phrases such as "Can you add another section to the body?".

3.5.2 Eliciting phrases based on AI capabilities. We focus on the AI capabilities *summarize*, *translate*, and extend in this section. We marked these rows in bold in the tables. As shown in Table 4, phrases for giving specific tasks to an AI had between 1 and 54 words, with a mean length per AI capability between 4.54 and 4.96 words. The motivations for delegating a summarization task to the AI were primarily efficiency and improving the text, such as "Summarise highlighted section" and "Help and summarize the meaning of this text, to ensure it is reflected in the first paragraph". In contrast, translations were motivated purely by efficiency (i.e. to replace the need for manual translation), as the coding of "goals" revealed in Table 16. When delegating text extension, participants were also motivated to gain inspiration, for example, with phrases like "I need more content on this". Coding of "styles", as shown in Table 15, revealed that participants largely used an imperative style (i.e. giving instructions). Finally, their phrases imply that the AI acts in the role of a co-author (see Table 17), for example "Write a longer alternative."

Table 5. Key findings from our survey (N=121), covering text editor usage, collaboration, and eliciting how people would delegate tasks to AI.

Key Finding	**Details**
Interactive comments are used for collaboration and task delegation	Inside documents, interactive comments are used by almost a quarter of participants for communicating about collaboration at least on a weekly basis. Tasks are delegated more frequently with interactive comments than document chat messages and inline text annotations.
Prompt lengths depend on writing stages and requested AI capability	Across writing stages, task delegations have a mean length between five and seven words. Delegations for the tasks summarize, extend, and translate had a length between four and five words.
AI is envisioned to act as a co-author	People prefer an imperative style for delegating tasks and desired an assisting AI across all writing stages. When outlining a text, they also assume AI to act akin to a co-author. For using an AI capability, participants refer to AI much like a co-author in task delegations.
Intent-driven functions	The survey collected concrete examples of how users would express their intent to access intent-driven writing functions in the functional space offered by an LLM. Based on the elicited samples we built an intent recognition component into our first prototype.

3.6 Survey summary

The analysis of our survey reveals how users encode conceptual functions through phrasing their writing intents. Concretely, we elicited how people would delegate writing tasks to AI. Since we conducted the survey in 2020, the elicited intent phrases are unbiased by experiences with conversational AI systems that became popular in 2022.

By collecting samples across writing stages, we captured a high-level picture of conceptual functions instructed to AI. Complementary, the phrases for delegating a certain task revealed insights on a detailed level per AI function.

In summary, across all writing stages, participants asked for AI assistance, but when asked for delegations for specific capabilities, they predominantly assumed an AI co-author. We interpret this as a general vision and need for assistance, combined with an expectation of specific writing support from AI co-authors. For our first prototype, this aspect motivated us to design a user interface in which the assisting AI appears akin to a "co-author" (e.g. via presence indicators and comments known from human-human collaboration).

Moreover, the coding of goals/motivation revealed how the high-level intentions for AI assistance depend on the writing stages: Participants looked for inspiration during outlining and for text improvements while drafting and revising.

Efficiency was an overall motivation for delegation to AI across all writing stages and specifically for translating and summarizing tasks. Beyond this, participants aimed for inspiration when extending text and also for text improvements when asking for summaries.

The intents expressed in the elicited phrases reveal users' expectations about AI-generated text (i.e. the output of conceptual functions). These expectations also relate to concrete parts of the text that users assumed to interact with (i.e. parts of the text they involved into conceptual functions), as also revealed by their phrases, such as implying different scopes, such as words, sentences, sections, the whole text. The mean length of the elicited task delegation phrases ranged between five and seven words across writing stages. Looking at the AI capabilities summarize, extend, and translate, the phrases had a mean length between four and five words. An overview of the key findings is given in Table 5.

Overall, this analysis supported us in understanding user intents prompted to AI, and the conceptual functions they are aiming to fulfil.

4 Prototype 1 - Conversational UI

We combined the findings from our survey with a design session in our research group to inform how to integrate AI capabilities into a text editor. This resulted in our idea of using interactive comments as annotations for AI, concretely, to integrate a conversational UI into these comments to delegate tasks to AI. Note that at the time of designing this prototype (2021), the conversational abilities of language models were far more limited than today, and conversational UI patterns like in ChatGPT or a "Copilot sidebar" did not yet exist (cf. [67] in 2022).

4.1 Design concept

We started with a brainstorming session as a group on a digital whiteboard, followed by sketching UI design ideas individually. Based on these sketches, we decided to follow the approach of integrating AI into interactive comments. While designing these features, we identified spatial, temporal, and informational design dimensions as cornerstones for designing a conversational AI. These design dimensions impact positioning, delays, and communication with the AI.

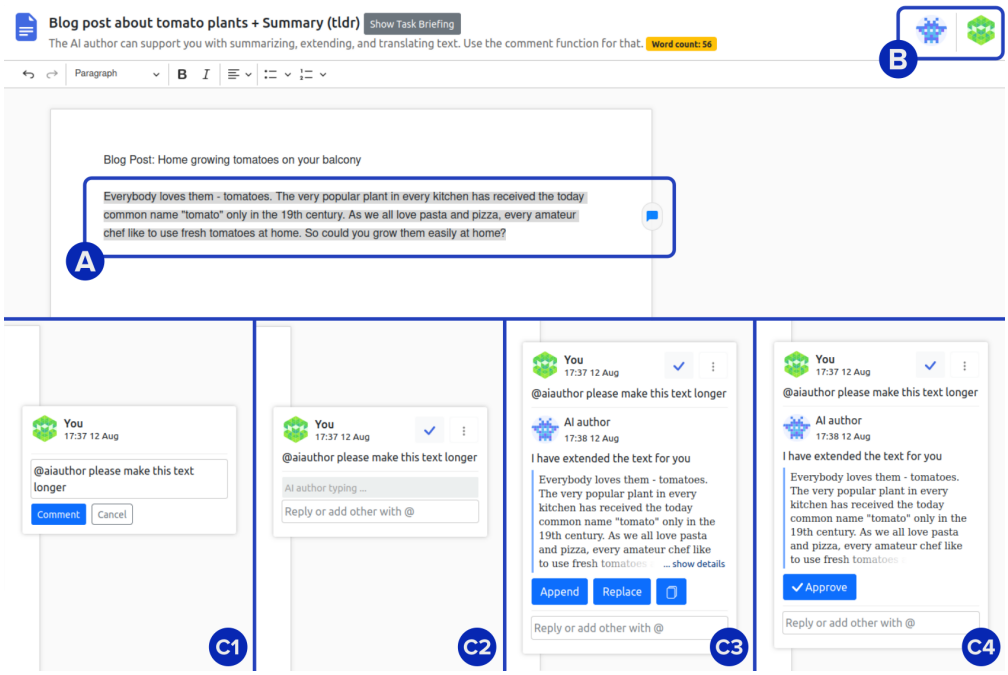


Fig. 3. Screenshots of our first prototype with a conversational UI. The top half of the figure shows the editor. We designed the editor to look similar to existing online text editing tools. Part A shows a user-selected text, with a balloon icon next to it on the right for adding an interactive comment. Part B shows that both the user and the AI are "present" in the document. The bottom half of the figure shows the procedure of delegating a task to the AI via an interactive comment. C1 shows a comment to extend the text. C2 shows an indicator that the AI is carrying out the task ("typing"). In the top right of the comment card, users can approve the comment, or use the dropdown to edit or delete the comment. C3 shows the AI's suggestion, three options to accept the text (append, replace, copy to clipboard). Furthermore, users can view more details and potentially write a reply. However, our system did not support multi-turn conversations. C4 shows the comment card after the users have clicked on "append" and the suggestion has been appended in the main text – users can then approve this result, which closes (removes) the comment.

The final design consists of four major parts and is shown in Figure 3. This figure shows (A) the text editor that allows for writing with AI. We integrated the AI similar to a human collaborator. (B) The AI is "embodied" in the top right of the screen with a user icon. This user bar signals the availability of the AI. The bottom half of the figure zooms in on interactive comments next to the page view. The comments (C1) allow for conversations, (C2) addressing users/AI (@aiauthor), (C3) presenting suggestions, and (C4) interacting with the suggestions, such as accepting and rejecting them. The comments displayed a reply input field, yet multi-turn conversations were not supported. Moreover, the comment interface provided three different options to work with the suggestion (appending it to the selected text, replacing the selected text, and copying it to the clipboard). It also offered a detail view for the suggestions, see Figure 18.

Using interactive comments as the central interaction element for AI has the benefit of providing a visual reference to the annotated parts of the document while being visually decoupled from it. Comments are also particularly used for task-oriented communication [9] by users. With such annotations, the interaction is less intrusive, is user-initiated, and lets writers stay as close as possible to their usual writing workflow [14]. Having the comment cards in the UI also served as a space for the AI to suggest changes [61] and for the user to track the history of such generated snippets.

Moreover, we designed the AI agent to behave akin to a human in its use of the comments. For example, an indicator showed when the AI "typed" an answer, and response times were randomly extended by 10 to 19 seconds.

4.2 Technical implementation

We prototyped the frontend with React[2], CKEditor 5[3], and Bootstrap[4]. We implemented a CKEditor Plugin for marking and referring to text. The intent recognition was based on the RASA framework[5]. Concretely, we trained a Dual Intent and Entity Transformer classifier with RASA in the default configuration (epochs set to 100) and used the phrases collected from our survey as the training dataset. We used this intent recognition to call the fitting AI capability from the backend, given a user's comment text. These AI capabilities were provided through separate REST APIs based on FastAPI[6]. The generative models were T5 for summarizing text, Opus-MT for translating text, and GPT-Neo for extending text.

5 Study 1: Collaborative writing with AI on a document through a conversational UI

In our first study, we investigated how people write collaboratively with an AI agent on a text document. In particular, we looked at how people integrate the AI into their workflow. Participants could use a conversational UI in an interactive comment to delegate tasks for extending, translating, and summarizing text to the AI. The study was a mixed-methods online experiment. Combining qualitative and quantitative methods, we gained a comprehensive picture of users' interactions and experiences. This investigation empirically contributes to how people express their intent through a conversational UI to access an AI function provided by an LLM.

5.1 Task

We designed the task to engage participants to collaborate with the AI for different aspects of writing. Concretely, we gave participants the task to write a blog post about growing tomato plants at home in English. We chose this topic since we consider it neutral. The topic might further require a short research phase, depending on individual familiarity with it. Participants were only allowed to use certain online resources. We provided them with a list of hyperlinks to existing guides and blog posts about tomato plants in German. Thus, this task provided a realistic opportunity for using translation. Participants had to finish the blog post within 30 minutes – thus, the task provided a potential incentive to try out AI text extensions. Furthermore, people were instructed to add a summary of three sentences about their blog post – thus, the task also included an opportunity to use AI summaries. Participants finished the task with a mean time of 40.01 minutes (SD=7.97, min=28.81, max=55.61).

5.2 Apparatus

Participants used our conversational UI prototype implemented as a web app, as described in Section 4.1. They used a web browser on their own computer to take part in the study and joined an online call with the experimenter.

[2]https://react.dev/
[3]https://ckeditor.com/
[4]https://getbootstrap.com/
[5]https://rasa.com/
[6]https://fastapi.tiangolo.com/

Fig. 4. The procedure of our online experiment took approximately one hour. After reading an intro text and giving informed consent, users read a task briefing that also included a video to introduce the features. Participants could ask for clarification. Then, their task was to write a blog post about growing tomatoes at home. We recorded the interaction logs, observations, verbal feedback, and the shared screen. This was followed by a semi-structured interview. In a final step, participants filled in a questionnaire with subjective ratings and questions about socio-demographics.

5.3 Participants

Ten people participated in our study (4 female, 6 male). Their age ranged from 27 to 36 years, with a median of 30. Five participants self-reported to speak English well, four very well, and one fairly well. Self-reports for German language proficiency showed nine native speakers and one person to speak German very well. All participants were recruited via social media and compensated with 12 € per hour.

5.4 Procedure

Participants joined a video call with the experimenter who provided them a hyperlink to a web application that guided them through the experiment. The experiment's procedure is depicted in Figure 4 and took about one hour.

In the first step, the web application presented an introduction about the study and data protection regulations, and participants had to give their informed consent. This step was followed by the task briefing (Section A.2). People read the instructions and watched an explainer video. They could ask the experimenter for clarification. In the third step, participants used the conversational UI prototype to complete the task. They shared the screen (web browser window) via the video call software, which we recorded. The experimenter asked questions of the semi-structured interview throughout the session, depending on user interaction with the tool and completed the interview after the task had been done. For the final step, the screen sharing and recordings were stopped, and participants filled in a questionnaire. It covered socio-demographics, language proficiency, and Likert items on usability aspects, including the SUS questionnaire [11].

5.5 Results

We analyzed the quantitative log data with a focus on how users interacted with AI through the conversational UI, and on timing as a key aspect of their broader editing behavior. We complement this analysis with qualitative findings.

5.5.1 Task delegation. We analyzed how participants delegated writing tasks to the AI through the conversational UI. They added a total of 139 comments, of which 131 were sent to the AI, while eight were removed without writing an instruction for AI. Each participant posted a mean of 13.1 comments (SD=4.41, min=5, max=17) with a mean length of 1.79 words (SD=0.89, min=1, max=3.8). In four cases, participants tried to combine AI skills by using the word "and" in the comment (e.g. "translate *and* extend"). Writing a comment took a mean time of 8.93 seconds (SD=12.52, min=2, max=119).

We counted 128 AI replies. Three comments were removed by the participants before the AI reply arrived. AI replies had a mean length of 52.96 words (SD=44.88, min=1, max=201). *Extensions* were the longest with a mean length of 90.21 words (SD=34.20, min=24, max=312), followed by *translations* with a mean length of 43.95 words (SD=52.74, min=1,

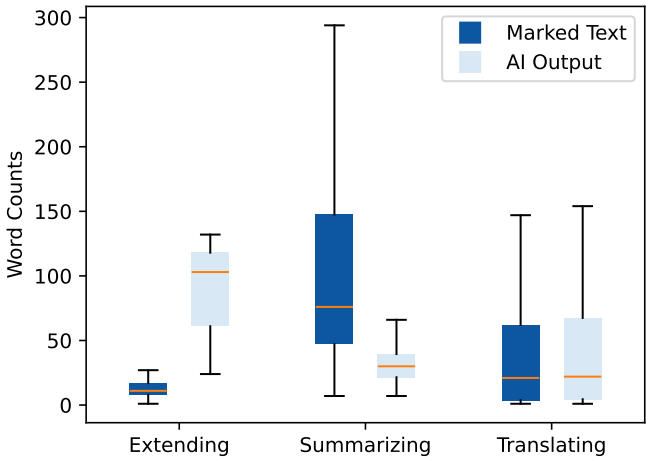


Fig. 5. Lengths of AI text generations and the lengths of the users' selected text, measured as number of words. This visualization shows how users employ AI functions to modify text with varying length characteristics: *Extending* is applied on short text selections, and the AI then generates text that is several times longer. With *summarizing*, input text length varies the most, and output is much shorter. The ratio for *translating* is close to one.

max=201). *Summaries* were the shortest with a mean length of 30.95 words (SD=12.88, min=7, max=66). Figure 5 shows these lengths of AI responses next to the lengths of the users' corresponding text selections. This figure reveals that users apply the AI capabilities to transform text in different ways. When extending the text, users select short snippets, and the AI generates text that is several times longer. In contrast, with summaries, longer input text is shortened considerably by the AI. With translations (German-English), the ratio is close to one.

The analysis of comment editing behavior showed that participants input a mean of 13.93 keystrokes per comment (SD=10.5, min=2, max=71). This excludes editing actions (backspace, clear, cut, delete). Analyzing these separately, participants corrected their input with a mean of 2.5 editing keystrokes per comment (SD=2.13, min=1, max=12). Eight participants deleted parts of their input with a mean sequence length of 2.72 editing keystrokes per affected comment (SD=0.75, min=2, max=4). These sequential edits happened in 18 comments.

5.5.2 Parallelism. In our prototype, the AI "writes" asynchronously. While it generates text, the user can still interact with the document. We looked into two types of this parallelism, which we call "short-term" and "long-term" parallelism.

Short-term parallelism: For short-term parallelism, we considered the window of time between a "comment posted" event and the corresponding "AI reply" event. This time includes our API's response time, plus our added delay to emulate writing (Section 4.1).

In total, eight users kept interacting during these timeframes, that is, while the AI was replying. They interacted in parallel in this way a mean number of 5.13 times (SD=4.55, min=2, max=15). The first such parallel action happened after a mean of 12.47 minutes (SD=8.30, min=3.28, max=29.30). We did not track reading behavior, but all participants might have read text as part of this parallelism as well.

Long-term parallelism: For long-term parallelism, we computed the window of time from a "comment posted event" until that comment's "closing" event, which happens when users actively close (delete or resolve) the comment. Everyone

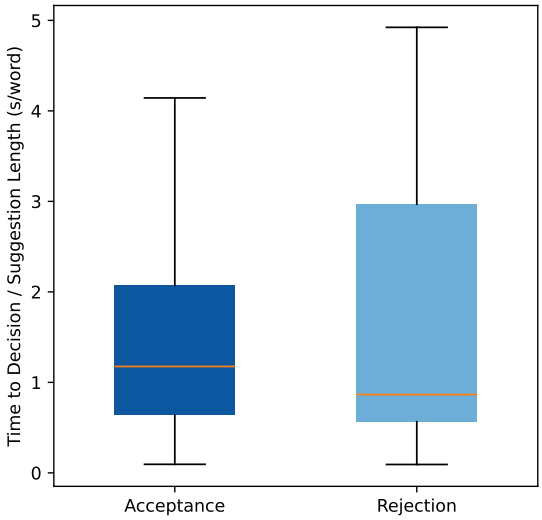


Fig. 6. The time participants took to decide whether to accept or reject an AI-generated comment, normalized by the length of the generated text. This plot shows that acceptances happened faster than rejections.

used comments in parallel in this way, that is, working on something else with at least one open comment, with a mean of 6.3 times (SD=3.92, min=1, max=13). The first such long-term parallelism happened after a mean of 9.60 minutes (SD=6.54, min=0.82, max=21.58).

5.5.3 Acceptance and rejections. We analyzed acceptances and rejections of AI-generated comments. Users could explicitly accept these suggestions via the three buttons *append*, *replace*, and *copy to clipboard*. We counted these interactions as acceptances.

Furthermore, participants could delete comments, which we count as rejections. They could also resolve a comment at any time, which has more nuance: We counted resolve events as approvals of the suggestions when users chose to integrate them by appending or copying the text to the clipboard beforehand. If no such accept events happened beforehand, we counted resolve events as a negative signal, adding to the rejection count.

Figure 6 visualizes the time that participants needed to arrive at a decision to accept or reject an AI suggestion, normalized by its length. On average, participants were slightly slower to accept suggestions, but the times for rejection varied more.

Acceptance: AI suggestions were accepted a total of 86 times, with a median normalized time to acceptance of 1.17 s/word (SD=9.45, min=0.09, max=82.5). Relative acceptance rates per AI capability revealed that translations were accepted most often with 83.33% (40 of 48), followed by summarizations with 65.11% (28 of 43), and extensions with 48.64% (18 of 37).

Rejection: Users rejected a total of 35 suggestions, with a median normalized time to rejection of 0.87 s/word (SD=20.53, min=0.09, max=112). Rejections per AI capability happened most often for extensions with 48.64% (18 of 37), followed by summarizations with 27.90% (12 of 43), and translations with 10.42% (5 of 48).

5.5.4 Final Texts. Overall, participants produced blog posts with a mean length of 386 words (SD=52.92, min=334, max=491). We computed the Flesch reading-ease score as a criterion for readability. The mean reading ease was "fairly easy to read" (72.04) and ranged from "Easy to read. Conversational English for consumers." to "Difficult to read." (SD=10.10, min= 49.93, max=84.88).

Table 6. Overview of the key findings from our user study (N=10) on writing with AI on a text document using a conversational comment-like UI.

Aspect	Key Finding
Task delegation	Participants used very short, command-like instructions for delegating tasks with a mean length of 1.79 words.
Selection behavior	Participants selected text depending on the desired AI capability (e.g. shorter selection for extending text, longer selections for summarizing text)
Short parallelism	While the system is executing the user's delegated tasks, participants read and work on different parts of the document.
Long parallelism	Multiple task delegations happen in parallel and AI outputs (suggestions) are often kept in the UI for a longer period of time.
Acceptance	Accepting suggestions happens slower than rejecting them. Suggestions were accepted more often if they fulfilled the user's intent.
Specified AI functions	Participants requested pre-determined AI tools, e.g. an AI toolbar confining the flexibility for accessing AI functions.

5.6 User perception: verbal feedback, observations, and ratings

To get a more complete picture of the interactions, we asked participants in a post-hoc questionnaire to rate Likert statements on usability aspects. An overview of the ratings can be found in Figure 16, and the questionnaire is available in Table 10.

Ratings on the SUS revealed a mean score of 83.5, ranging from 67.5 to 92.5, see Figure 16a. This score is equivalent to an excellent usability [4]. Furthermore, the majority of participants found the AI system natural (3 agree, 4 strongly agree), see Figure 16b. Also, the majority agreed that the tool made them reach their goal faster (3 agree, 5 strongly agree) and supported them to complete their task (5 agree, 3 strongly agree).

The experimenter took notes with the approximate timestamps while participants worked on the task. These notes captured surprising or interesting observations of the participants' interactions and comments. The notes were categorized, for example, as reactions to the AI capabilities, the participants' overall editing strategy, and their parallel workflow. Another researcher of our group reviewed the recordings, notes, and categories. In a brief discussion, we clarified and corrected unclear notes.

These categorized observations and responses to the semi-structured interviews revealed four important themes that we elaborate on next.

5.6.1 Conversations with AI vs. AI as tools. Four out of the ten participants tried to converse with the AI by trying to reply to a comment. P5 suggested further conversation could be used to switch to another AI capability if intent recognition failed. In contrast, three participants (P2, P3, P5) in the interview requested to put AI into tools (e.g. buttons) rather than delegating tasks with natural language.

5.6.2 Parallel workflows. The analysis of the interaction logs revealed parallelism in conversations with AI (Section 5.5.2). We also noted this in our observations for six participants. These participants not only edited the text in parallel to the AI executing tasks, but they also used that time to read their text (P4) or to go through other text material for researching facts (P2, P7, P8). Participant 4 reflected on this in the interview: "the delay feels good, especially if I do things in parallel. However, skipping between parts of the text is hard (if capabilities run in parallel). But if I only want one thing to happen, instant execution might be better". Participant P8 responded, "At first, I found it annoying, but it would also take some time to write if it was a partner. Faster is better, but if it does not disturb [you], you could also do different things in the meantime."

5.6.3 Mental models of AI and interpretation of its inner workings. For six participants, their comments revealed that they reasoned about how the AI worked internally. In particular, factors such as delay and context influenced their interpretations, as reactions show, such as by P6 "It takes longer, that must mean something good.", or P10 "I thought it would be faster for shorter text", and P7 "I thought the AI reads the thing [part of the text] and wouldn't need keywords as input". Some participants also attributed more general, almost human-like behavior and skills to the AI, such as P2 reacting surprised about a generation, "cool, it's googling for me", as well as P9 "oh he thinks, that I have started in June, interesting".

5.6.4 Specific use of AI and writing strategies. All participants used the AI capabilities in specific ways and integrated them into their writing strategies. For example, this included translating single words (P2, P9, P10) and extending partial sentences (P3, P7, P10). Users also adapted their overall writing strategy with AI. For example, P5 started with manual writing and intended to apply AI functions later, while P4 took another approach by translating a long text from the sources and then summarizing the resulting paragraphs manually.

5.7 Conversational UI summary

We investigated how users edit text documents in collaboration with an AI agent through a conversational UI. Our findings emphasize how users express their intent through a conversational UI to access a function provided by LLMs. Such models have evolved and become more powerful since we conducted the study in 2021 but most recent UIs have not yet realized the implications in line with our participants' behavior and feedback. In particular, from a user perspective, our results support to further investigate the emerging research direction towards direct manipulation [38]. We summarize the findings below and provide an overview in Table 6.

Our prototype offered a high degree of functional flexibility in accessing AI capabilities by instructing user intent in natural language. However, participants accessed AI functions through short, command-like instructions, and corrected these instructions only minimally. Furthermore, text selection behavior varies with intended AI functions. For example, when extending text, users select short snippets, but when summarizing text, users select longer input text.

Participants worked with AI in parallel. While the system generated an output, people carried out other tasks such as writing and reading. They also initiated multiple AI generations in parallel and kept multiple resulting AI annotations open in the UI. With this approach, they enhanced their perceived efficiency and kept track of AI suggestions.

These suggestions were more often accepted than rejected. Acceptances needed slightly more decision time relative to the suggestions' lengths than rejections.

In their verbal feedback, participants requested an AI toolset. They asked for lowered flexibility in accessing functions provided by LLMs. This feedback aligns with the analysis of their comments that revealed short command-like instructions for delegating tasks to AI. These findings pointed to pre-determined AI tools, asking for confined UIs for

accessing specific AI functions. This feedback motivated us to iterate on the prototype. Our second prototype then provided AI capabilities as part of a toolbar instead of a conversational UI.

6 Prototype iteration towards a toolbar UI

The findings from our first study informed how we adapted the prototype for our second user study. Summarized on a high level, we replaced the conversational UI with a floating toolbar UI, which included buttons for predefined AI functions, plus free prompting. That way, we offered two ways in accessing AI functions. With the toolbar UI, we restricted the flexibility, while offering a maximum of flexibility in accessing the AI functions.

6.1 Reflection on the design implications of study 1

Participants in our first study wrote short commands for delegating tasks to the AI. This included single-word commands, such as "translate", "extend", or even just "ext". Furthermore, participants requested to use the AI functions through buttons. For example, P5 suggested to "put the functions into buttons or floating tooltips", P3 would rather "use the functions without the comments", and P2 mentioned "AI capabilities could have been implemented into the context menu". Compared to the elicited task delegations in our survey (Section 3), users thus "minimized" their input for delegating tasks to AI during actual interaction in a text editor. Concretely, the mean length of inputs in our first study was 1.79 words, while in the survey it was between 4 and 5 words. Thus, both qualitative and quantitative results pointed us into the same direction: People preferred efficient commands over open delegation prompts. Interestingly, in retrospection, participants' behavior and wishes prior to exposure to ChatGPT thus went against the recent product trend of adding very open prompting features in the UI, which assume elaborate prompting (e.g. chatbot assistants in sidebars).

6.2 Changes to the UI

Based on these findings, we decided to replace the conversational UI with a toolbar UI. This toolbar offered the three functions *extend*, *translate*, and *summarize* as buttons. Furthermore, it offered a button that opens a text entry field for prompt input. At the time (early 2023), such prompting had become widely available via systems such as ChatGPT. This motivated us to include it here, thus offering both fixed AI functions and open prompts.

As a part of introducing the toolbar UI, we also changed the text highlighting/markers. In our new prototype, we used different colors, one for each AI function (extend, summarize, translate, free prompt). Furthermore, we adapted the detail view to a column layout that displayed the user's selected text on the left and the system's suggestion on the right, to ease comparison (Figure 19).

6.3 Changes to the system responses

We changed the system response as well. In the first prototype, we used comment-card elements floating at the right edge of the document page to include the conversational UI. We kept these cards for the second prototype but with reduced interactivity. Comments were completely removed, such that the cards only display the AI response. This new design is shown in Figure 7. The figure shows (A) the floating toolbar UI offering pre-determined AI functions and a fourth button to access free prompting. (B1) shows a loading indicator that represents the AI executing the task, and (B2) shows the AI's suggestion, an icon to show the details, and options for accepting the suggestion. For *extend*, the generated text could only be appended or copied to the clipboard. That is, we removed the "replace" button since this would have overwritten the user's original text, which does not match the user expectation for *extend*. (C1) shows

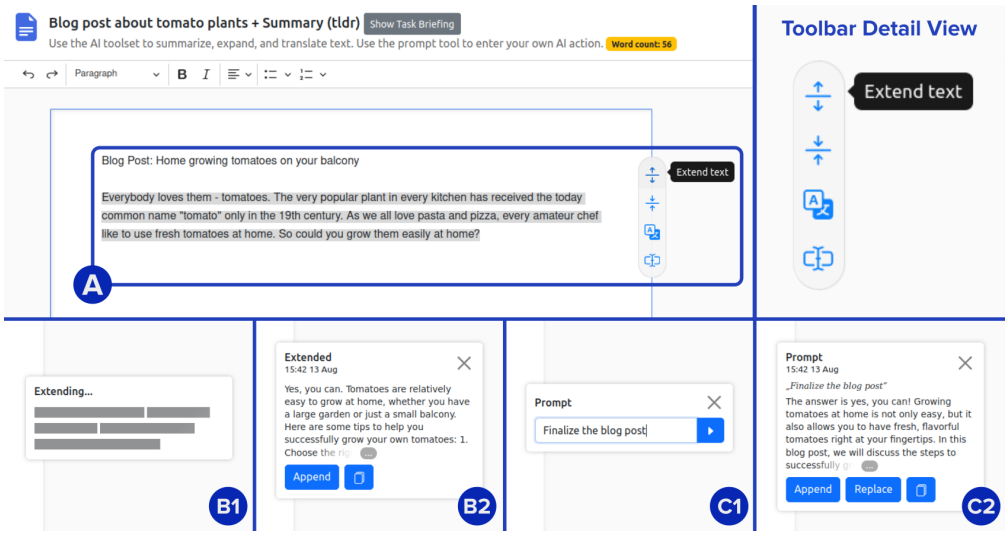


Fig. 7. Screenshots of our second prototype with an AI toolbar. The top half shows the editor, similar to existing online text editor tools. Part (A) shows how users could select text, which brings up the AI toolbar floating next to it on the right. This toolbar provided buttons for extending, summarizing, translating, and prompting own functions. The bottom half shows the procedure of applying an AI function from the toolbar. (B1) shows a loading indicator after a user has clicked on the extend text function. (B2) shows the AI's suggestion, with buttons for viewing the complete suggestion ("..."), appending it, and copying it to the clipboard. Replacing (cf. Figure 3) was not possible for "extend" since it would have overwritten the source text. (C1) shows the UI for prompting own functions. (C2) shows the AI's response to this.

the interface for free prompting, and (C2) shows the corresponding AI response. The cards could be closed (removed) through the cross icon in the top right.

6.4 Implementation changes

We switched the model to extend text from GPT-Neo to GPT3, and used this model also for executing custom prompts. The models for summarizing and translating text remained the same.

7 Study 2: Writing with a toolbar UI with pre-determined AI tools and the option for custom prompts

The design for our second user study remained the same as in study 1 (Section 5). The only change was the prototype. We reified writing intents in pre-determined tools and also offered free prompting of an LLM as an alternative. Offering these options for accessing AI capabilities, we varied the degrees of functional flexibility in the UIs. The pre-determined AI tools offered a minimum of functional flexibility, and the free prompting a maximum of functional flexibility.

7.1 Participants

We recruited 12 participants (5 female, 7 male). Their median age was 28.5 years (range: 20-36). Five self-reported their English language proficiency as very well, another five as well, one as fairly well, and another one as poor. Eleven people were German native speakers. One rated to speak German well. All participants were recruited via social media and compensated with 12 € per hour.

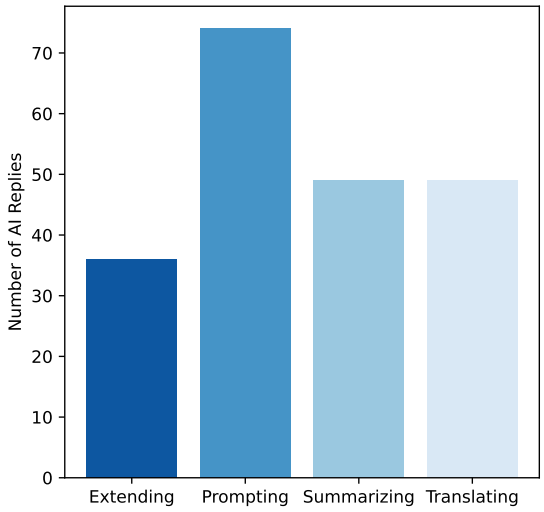


Fig. 8. Use of the AI tools in our second user study. Free prompting was used most often, followed by pre-determined tools for summarizing and translating. Extending text was used the least.

7.2 Results

Similar to the first user study, participants finished the writing task with a mean time of 39.57 minutes (SD=9.01, min=22.85, max=51.48).

7.2.1 Data cleaning. In total, participants used the AI tools 243 times. Of these, 35 events had an empty AI response or no response at all.[7] We thus excluded these 35 events and considered the remaining 208 usages of AI tools for our analysis.

7.2.2 Use of pre-determined AI tools and free prompting. On average, participants used AI tools 17.33 times (SD=7.0, min=7, max=30). They prompted a total of 74 own functions (35.57% of AI uses), while the summaries and translations were used 49 times each (23.56%), and text extension was used 36 times (17.31%), as shown in Figure 8. Only one participant tried to combine AI functions by using the word "and" in the prompt, namely "integrate both sentences *and* shorten thereby".

Each participant prompted a mean of 6.17 own functions (SD=2.82, min=1, max=11) with a mean length of 5.59 words (SD=2.93, min=1.5, max=10.33). AI responses on these free prompted functions had a mean length of 59.82 words (SD=45.92, min=2, max=205), overall AI responses were shorter with a mean of 56.39 words (SD=45.43, min=2, max=205).

Figure 9 shows the characteristics of the input and output lengths. For extend, summarize, and translate, these are similar to the results in study 1 (see Figure 5). Freely prompted functions make the text longer on average, and the lengths of the resulting responses vary less compared to the input.

We also analyzed the editing behavior when prompting own functions. Participants input a mean of 37.88 keystrokes per prompt (SD=27.62, min=2, max=126). This excludes editing actions/keys such as backspace, clear, cut, and delete.

[7] Note that at the time of conducting the study, the OpenAI web API was under high load and randomly timed out.

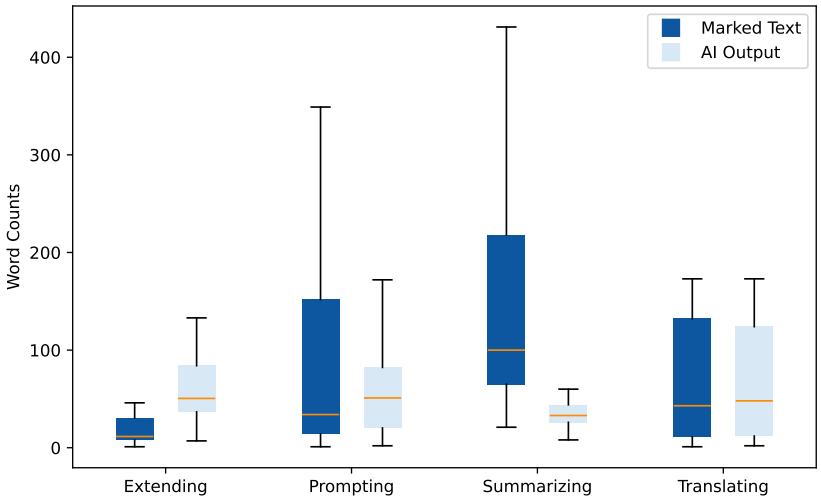


Fig. 9. Paired lengths of AI generations and user-selected text, measured as number of words. With free prompting of functions, AI output resulted in shorter text compared to the original, selected text. The characteristics of the other AI tools remained similar to our first user study (cf. Figure 5.

These editing actions/keys had a mean of 4.76 keystrokes per prompt (SD=4.3, min=1, max=17). Nine participants pressed editing keys in sequence. They sequentially deleted input with a mean of 3.49 delete-keystrokes per prompt (SD=1.57, min=2, max=7). This sequential editing occurred 21 times.

7.2.3 Parallelism. Similar to the first user study, we analyzed short-term and long-term parallelism.

Short-term parallelism: Two participants carried out parallel interactions while the AI was replying. Each of them did so once, concretely, while waiting for the response to a free prompt input.

Long-term parallelism: As in study 1, we computed long-term parallelism as interactions that happened between triggering an AI tool and removing the resulting card. Cards were used in parallel by all users, with a mean of 3.67 times (SD=2.71, min=1, max=10). Users began to use cards in parallel after a mean of 16.94 minutes since the start of the task (SD=10.02, min=0.87, max=33.57).

7.2.4 Acceptance and rejection. As in the first study, we analyzed acceptances and rejections of AI-generated text. Similar to the first version of our prototype, users could accept an AI generation by appending, replacing, and copying it to the clipboard. We counted these interactions as acceptances. We counted rejections when closing cards without any such acceptance interaction. Figure 10 shows the decision times for acceptances and rejections, normalized by the length of the suggestions. As in study 1, participants accepted suggestions faster than rejecting them.

Acceptance: A total of 137 AI generations were accepted. The median normalized time to acceptance was 0.29 s/word (SD=1.21, min=0.03, max=12.67). Separated by functions, translations were accepted the most with 87.76% (43 of 49), followed by prompted functions with 67.57% (50 of 74), extensions with 58.33% (21 of 36), and summarizations with 46.94% (23 of 49).

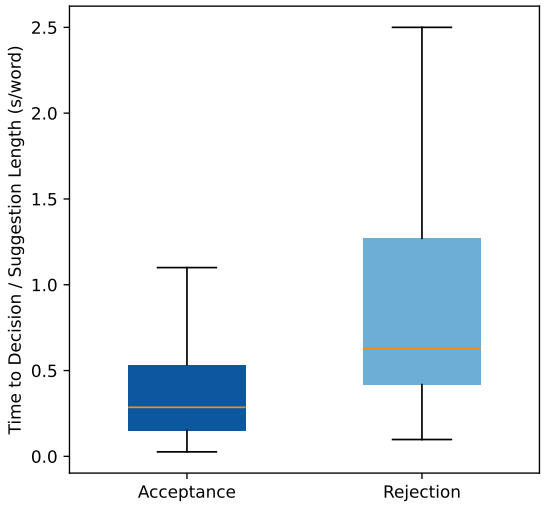


Fig. 10. Visualization of the time that users spent on deciding to accept or reject AI-generated text. The times are normalized by the lengths of the generated texts. This plot shows that acceptances happened faster than rejections. Rejections might have taken people more time since they might be cognitively more demanding.

Rejection: Participants rejected a total of 56 AI suggestions. The median normalized time to rejection was 0.63 s/word (SD=2.62, min=0.09, max=61.5). People most often rejected summaries with 48.98% (24 of 49), followed by prompts with 27.03% (20 of 74), extensions with 22.22% (8 of 36), and translations with 8.16% (4 of 49).

7.2.5 Final Texts. Participants created blog posts with a mean length of 339.83 words (SD=29.59, min=304, max=410). The Flesch reading-ease score showed a mean readability of "Plain English" (68.75) and ranged from "Easy to read" to "Fairly difficult to read." (SD=7.97, min=56.76, max=81.43).

7.3 User perception: verbal feedback, observations, and ratings

As in study 1, we analyzed the qualitative data to learn about participants' perceptions when interacting with the prototype. They rated Likert items on usability via the a SUS questionnaire and additional items on topics. An overview of these ratings can be found in Figure 17. The questions are listed in Table 11.

The mean SUS score was 87.29 (SD=6.94, min=77.50, max=97.5), see Figure 17a. This score reflects an "excellent usability" [4], and is above the score of our first prototype. Figure 17b shows the additional ratings, which revealed that almost everyone found the interaction with the system natural (7 agree, 4 strongly agree). Furthermore, all participants found that the system supported them in their task (3 agree, 9 strongly agree). Ratings on efficiency were similarly positive (4 agree, 7 strongly agree). In contrast to our first study, we further asked about control and authorship. A majority of participants agreed to be in control of editing the text during the task (5 agree, 4 strongly agree). Ratings on feeling like the author of the text were ambiguous (4 agree, 1 disagree, 3 strongly disagree).

Since tools such as ChatGPT became popular shortly prior to conducting our second user study, our questionnaire asked if participants had prior experience with intelligent writing tools. All indeed had such experiences: Eleven had experienced auto-correction and eight had used ChatGPT for writing.

We recorded each session and took notes of observations as in the first study. Data processing also stayed the same.

Based on the participants' feedback and our observations, we identified four themes which we describe in the following sections.

7.3.1 Users were unsure about prompting and wished for inspiration. Eight of twelve participants mentioned that they were unsure about prompting. For example, P4 was not familiar with it. Others struggled to come up with ideas for prompting – as commented by P5: "intuitively, I have not got an idea, extend already exists". Responses in the semi-structured interview revealed that they would have wished for examples of prompts, such as P7: "I wish for a list of examples, I was not sure if I was free about text input." Similar, P9 noticed "I used only primitive commands" and expected that a set of examples would allow them "to use it [prompting] more flexible". We provided participants with an initial example in the briefing and they were free to input and try out their own prompts. However, it seems this freedom hindered them at first, as also reflected on by P12. A reason for this challenge might be the creative and open nature of such prompts. P6, for example, first liked the translate feature the most but then favored the prompting of own functions since "it is the most creative". However, P4 perceived this as a demand ("you have got to be very creative") and P9 reflected: "I ask myself, how I can prompt more creatively". Guidance for prompting might support lowering such creative barriers; as P12 put it "give examples for prompting to lead the user".

7.3.2 Users adapted existing functions to improve quality and control over the output. We built our system by combining several models to provide the AI functions, for example, translations were provided by Opus-MT, summaries by T5, and GPT3 was used for extending text and free prompts. These models showed varying output quality. All participants gave feedback on how they perceived this. The extend function's output quality varied. For example, P1 found the output too vague: "not the story I wanted to go for [...] America is too general [as the origin of tomato plants]." On the other hand, P3 found the extend function useful: "I am surprised about the length, actually it is good". For the AI summaries, participants remarked on lower quality. P8 found that the "summary is unsatisfying" and P10 "that is not a good summary".

When expectations about output quality were not met by the pre-determined AI tools, participants prompted replacement functions. For example, six participants prompted their own summary functions. At the same time, some of them used the free prompting to further control their replacement function. For instance, P3 said that they are "prompting for summary since then I can be more specific". In the same line, P9 remarked "the length of the summary should be adjustable", followed by specifying the length of the summary in an own prompted summary function. Thus, prompting own functions in this way added flexibility and allowed users to specify custom versions of the existing AI tools.

7.3.3 Chaining multiple AI tools. Our observations showed that participants chained the different tools to reach their goals as part of their writing strategy. P5 and P6, for example, first translated German text into English and then extended the text. P8 prompted a function and then summarized the rather long result to shorten it. P9 combined various functions to modify parts of a text. This specific use of AI tools shows that participants utilize AI tools as "atomic" functions and combine them to reach an intended goal.

7.3.4 Mental model about AI and interpretations of the context. Even with the toolbar UI instead of conversations, participants reasoned about the system's inner workings, similar to our first prototype. They also interpreted the context of generations. For instance, P1 asked "is only selected context of interest?" and P4 interpreted the context more broadly when extending text with AI, saying it "[...] adds more detail. I found that also in the sources. Now, I

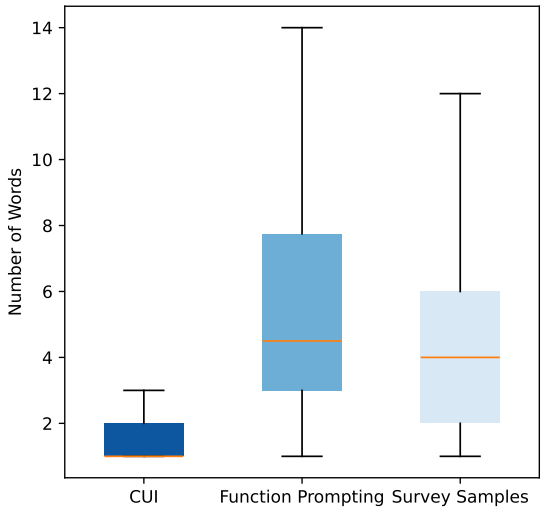


Fig. 11. In study two, participants used the free prompting tool with longer prompts compared to the text that participants in study one entered with the comment feature. The prompts reached lengths similar to those of the task delegation samples envisioned by participants in our survey.

know where that came from." Similarly, P5 interpreted the AI tools to involve external sources: "oh, he did research for me, it is a good support for research". This comment also reveals anthropomorphism ("he"), although we had removed the "embodied" representation from study 1 (e.g. no user icon for the AI). Reactions by P6 and P9 also showed this: When the API took a few moments longer to respond, P6 said "well, it seems he has to think about that", and again later. Similarly, P9 commented, after extending text, "Only one sentence output. It does not want to extend!". Uncertainty about how the AI tools work and generate text might make users think that AI tools have their own (human-like) behavior, similar to an agent-like presentation.

7.4 AI toolset & free prompting summary

In our second user study, we investigated how users edit text documents with pre-determined AI tools in a toolbar, as well as free prompt input. With our modified prototype, we varied the degree of functional flexibility offered by the UIs for accessing AI functions. The pre-determined tools offered a confined flexibility while the free prompting offered a maximum of flexibility. An overview of our findings is given in Table 7.

Most participants were unsure about how to use free prompting at first, although 75% had experience with tools such as ChatGPT. Nevertheless, after some exploration, participants applied self-prompted functions, for example, to create improved versions of already existing AI tools (summary) or to apply custom functions. In general, participants used AI tools as "atomic" functions and combined and chained them to reach their goals.

Their prompts entered via the free prompting box were longer than the conversation prompts entered in study 1, as shown in Figure 11. This indicates that free prompting of functions gives more flexibility to users than delegating a limited set of functions (summarize, translate, extend), but people have to put more effort into specifying their intent. Interestingly, the length of these freely prompted functions was similar to the length of what people had envisioned to enter as task delegations in our survey.

Table 7. Key findings from our second user study (N=12) on working with AI on a text document by using a toolbar UI that offers pre-determined AI tools as well as free prompting.

Aspect	Key Finding
People combined AI tools and free prompting	Pre-determined AI tools (extend, summarize, translate) accounted in sum for 64.42% of all triggered AI functions. Participants prompted their own functions in 35.57% of all uses of AI. These prompts had a mean length of 5.59 words.
Parallelism	Short-term parallelism from study 1 almost disappeared since AI tools executed without delay. Participants integrated AI tools more tightly into their writing workflow. Long-term parallelism was used by all participants, i.e. they kept multiple suggestions in the UI at the same time.
Acceptance	Accepting suggestions happened faster than rejecting them.
Varied degrees of functional flexibility	Our modified prototype offered two ways of using AI, varying the functional flexibility (AI toolset: minium flexibility, free prompting: maximum flexibility). The results show that he functional flexibility accounted for in UI designs affects how people integrate into their workflow.

Parallel workflows decreased for AI tools. In the first prototype, the AI response time was extended to emulate human writing behavior. Removing this delay decreased the overall response time, and participants integrated AI tools more tightly into their writing workflow.

Text generated by the AI tools was accepted in the majority of cases and deciding to do so took less time than deciding to reject it. Translations were accepted most often, followed by functions prompted by users. However, prompted functions were rejected second most often while almost no translation was rejected. The usability scores of the SUS questionnaire showed an increased usability of the new prototype.

Overall, the results of our second study show that functional flexibility accounted for in UI design affects how people integrate AI into their workflow. Our experience and findings from our survey and the two user studies motivated us to conceptualize the *functional space and functional flexibility* as a theoretical construct.

8 How user interfaces shape access to the functional space of generative AI

Here, we bring together the reported findings of our empirical investigations with contributions from related work to introduce the theoretical construct of *functional space and functional flexibility*. A theoretical construct for thinking about text-based interaction with generative AI: Its key concept is the *functional space* offered by LLMs to users, an unlimited space of output generations that fulfil conceptual functions. Such *functions* serve a user's goal. They might come with rather clear expectations, such as translating and summarizing a text, but might also be more fuzzy, such as inspiring and reflecting. *The role of the UI* is to determine how users can express their intents (e.g. click a summary button vs. enter a prompt "summarize this") and to which degree they can access their desired *target function* within this functional space. Through this conceptual lens, we describe how different UIs make AI functions accessible in different ways. An overview of these core aspects is shown in Table 8.

Following Generative Theories of Interaction [6], we have compiled a set of questions that can be asked to interpret the analytical, critical, and constructive power of functional flexibility as realized in a system, see Table 9. These

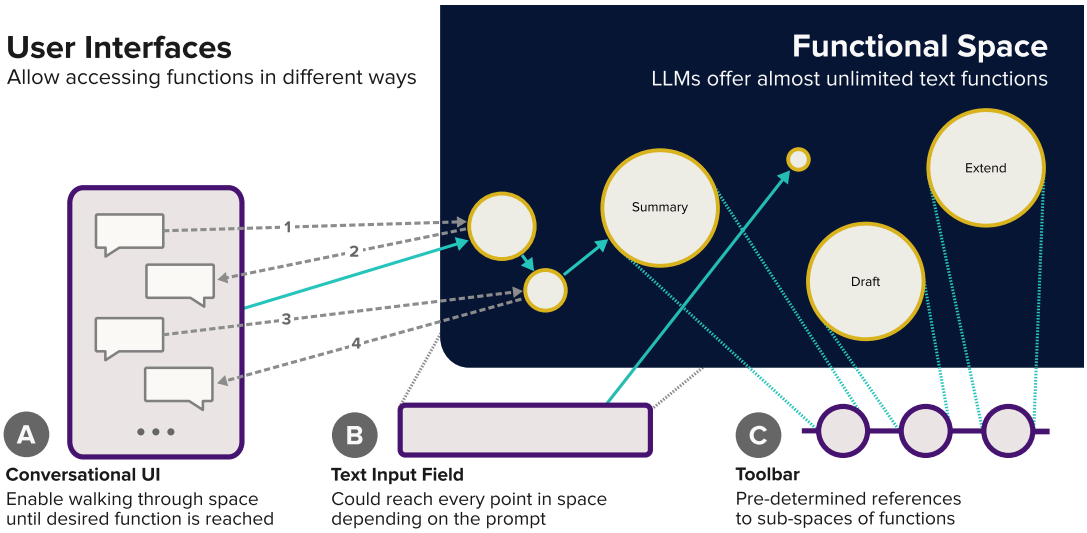


Fig. 12. High-level illustration of how different degrees of functional flexibility arise from UIs that make the functional space of generative AI accessible to users: The dark plane visualizes the functional space provided by an LLM. To keep this figure generalizable, the space is given no explicit dimensions. Subspaces within this space represent different text functions that users desire to access, i.e. their target functions, such as "summary". Each point represents a specific function (e.g. a specific way to summarize). The illustrated UIs are examples of actual UIs that allow users to dive into the functional space at different locations and limit the reachable subspaces: (A) With a conversational UI, users typically "walk" through the space via messages until a desired function is reached in conversation. (B) With a text input field for free prompting, users could directly reach every point in the space, depending on the prompt. (C) A toolbar offers pre-determined references to subspaces of functions. Overall, these UIs vary in the amount of involved context that is passed to the LLM, the possibility to prompt instructions, and subsequent user interactions on the LLM's responses. Our theoretical perspective emphasizes the role of the UIs as the essential connection between user intent and the functional space of LLMs. UIs define both (1) how users can express their intents and (2) to which degree they can access the LLM's space of functions.

questions help HCI researchers evaluate and refine AI systems, guiding the design of interfaces to offer the right degree (or mix) of functional flexibility.

In the remainder of this section, we take an analytical approach to examine our own applications through the lens of functional flexibility to address the question of "Can existing systems be deconstructed in terms of functional flexibility?" (cf. [6]). The discussion section then builds on this analysis, offering a critical and constructive use of functional flexibility with our prototypes as examples, and outlooks on applications to other systems (Section 9).

8.1 Grounding the theoretical construct of functional flexibility

In contrast to smaller, more specific models built for a certain task, the development of language models went towards general-purpose LLMs [12, 44] that serve a multitude of tasks. Such tasks fulfil conceptual *functions* that are based on user intents. Examples of intent-based functions in the context of text editing include paraphrasing, summarizing, or drafting a text [41]. Thus, general-purpose models offer a large *functional space*, that promises users almost unlimited possibilities of output generations. Users explore these possibilities by instructing LLMs from an similarly unlimited input space of written text. This view is related to other recent conceptual HCI work: Reynolds and McDonell [47] described interactions with LLMs as locating an already learned task (i.e. locating an already existing function). Finding

the right words to locate such tasks is termed "abstraction matching" by Liu et al. [33] and identified as the "gulf of envisioning" by Subramonyam et al. [53] (i.e. describing a function). According to Beaudouin-Lafon [5]'s theory of instrumental interaction, an LLM can be both: a tool, for example for editing a text document, but it can also be an object of interest, for example when engineering prompts for an LLM. In this work, we conceptualize the LLM as a logical component as part of a tool and extend prior perspectives on AI tools [48] by examining how UIs shape users' access to AI functions.

We combine these ways of thinking about interactions with LLMs, based on our reflections across the presented empirical projects, to extract the key insight that interacting with LLMs is about *accessing a desired target function* through suitable UIs. For example, prompt-based UIs expect from the user the capability to express their intents effectively in natural language. These UIs present a free text field for entering prompts, as implemented in our second prototype, see Figure 7 C1. The freedom to define any user intent, and the need to define it well [53], makes it difficult for users to access the right function within the large functional space offered by LLMs. In contrast, toolbar UIs offer pre-determined AI functions to users and thereby ease access at the cost of restricting the flexibility, see Figure 7 A.

Table 8. An overview of the core aspects of our theoretical construct on how user interfaces shape access to the functional space of generative AI.

Aspect	**Theoretical approach**
Functional Space	LLMs offer great functional flexibility which promises users an unlimited space of output generations that fulfill desired functions. Thus, an LLM can be considered as a functional space. Users instruct LLMs with text inputs from a similarly unlimited input space to access these functions.
Role of User Interfaces	UIs define how users can express their intents and to which degree they can access the LLM's functional space. Concretely, UIs allow diving into the functional space at different locations and limit the reachable subspace.
Key Insight	The key mechanism by which "interaction with LLMs" can be construed is accessing a desired target function through suitable UIs.

Table 9. Questions that can be asked to interpret the analytical, critical, and constructive power of functional flexibility implemented in a system [6].

Perspective	**Questions**
Analytical	What UI elements are available with which degree of functional flexibility? What mix of functional flexibility is offered by the UI / interaction design overall?
Critical	How does the degree of functional flexibility influence the interaction with the system? How does the offered degree of functional flexibility improve the system? Does the system/UI offer the right degree of functional flexibility to the user for reaching/identifying their goals? What problems does the degree of functional flexibility offered by the system cause? Can users reach/identify their goals with the degree of functional flexibility offered by the system?
Constructive	Can functional flexibility inspire new ideas when designing a new system? (Inspired by [36]) How can users be supported to reach their goal? How can we help users identify their goal? Which UI elements could be added to a design to increase/decrease functional flexibility?

Our perspective on the flexibility offered through UIs to access conceptual functions in the space offered by LLMs is similar to identifying creative concepts in a conceptual space, as introduced by Wiggins [63]. Thus, we define our concept formally based on Wiggins' formalization [64].

First, we define key aspects of this formalization:

U_{LLM}: The universe. A functional space containing all possible pieces of text.
$[[.]]$: An interpreter for selecting text pieces from U_{LLM}. This is the API of an LLM.
$<<.,.,.>>$: Another interpreter for traversing U and its subsets. This is the user.
R_P: Rules defining a function. These are instructions to the LLM, for example a prompt and the context.
T_{UI}: Rules for traversing the functional space. This is the UI, as it determines how a user explores the space of an LLM.
E_I: Rules for evaluating a function. Users evaluate if a function's output matches their intent.
$F = [[R_P]](U_{LLM})$: A particular functional space. This is a subspace of the almost infinite functional space.

Then, we define the process of exploring the functional space of an LLM as follows:

$$f_{out} = <<R_P, T_{UI}, E_I>>([[R_P]](U_{LLM})) \tag{1}$$

Through passing instructions R_P to an LLM U_{LLM} (directly through prompts, or indirectly through pre-determined instructions and interaction) , a user accesses the functional space F. The user $<<.,.,.>>$ explores this functional space through the UI T_{UI} and evaluates if the output matches their intent E_I. The output results in functions f_{out}, respectively a set of text generations fulfilling conceptual functions.

Thus, our theoretical construct of functional flexibility can be expressed as:

$$<U_{LLM}, [[.]], <<.,.,.>>, R_P, T_{UI}, E_I> \tag{2}$$

an exploratory system [64] of conceptual functions offered by LLMs to users.

In summary, we capture our concept in the new theoretical construct of *functional space and functional flexibility*. This provides a conceptual lens that highlights the role of user interfaces for LLMs and how they shape users' access to functions provided by LLMs. Our theoretical construct puts fresh emphasis on the role of UIs as the essential connection between the users' intents and the functional space of LLMs. In this way, UIs define both (1) how users can express their intents and (2) to which degree users can access the LLM's space of functions. Together, this determines the *degree of functional flexibility* made available to users.

8.2 Deconstructing how UIs make the functional space of generative AI accessible to users

Figure 12 visually summarizes how UIs make the functional space of generative AI accessible. The functional space provided by an LLM is visualized as a plane without specific dimensions to serve generalizability. Hypothetically, dimensions could be labelled depending on use cases, for example, sentiment [26], style [52], or could be dynamically derived from prompts [54]. Regions within this space (subspaces) represent different text functions that a user might want the model to execute. We decided to use subspaces, not single points, to reflect that many text generation systems are non-deterministic and text transformations can be underspecified. For example, if a user thinks of an intended function as "summarize", there might be more than one reasonable mapping, that is, multiple functions (and thus

outputs), given the same input. The sizes of these subspaces thus reflect the variety in how the model carries out the user's desired functions.

The UIs illustrated in the figure show how they allow users to dive into the space at different points and also limit the reachable subspaces that can be explored by the user with that UI. As concrete examples, we illustrated conversational UIs, text input fields for free prompts, and toolbars. These approaches differ, for example, in the amount of involved context that is passed to the LLM, such as conversation history, the possibility to prompt instructions, and subsequent user interactions that involve the LLM's responses. These differences shape how these UIs support the user in accessing a desired function within the functional space. Based on the deconstructed UI examples in the figure, we analyze our UI implementations in light of functional space and functional flexibility:

Example A shows a conversational UI, similar to existing industry chat tools such as ChatGPT, that offers a step-wise navigation of the functional space. Since the history of the conversation serves as context, with each utterance, the user iterates on the accessed function and thus pushes through the functional space until a satisfactory target function is reached. In our first prototype, multi-turn conversations were not implemented, yet some people asked if it was possible to reply to the AI, to iteratively push to other (refined) functions, for example, to adjust generations by making them longer ("do I have to write 'needs to be longer'?") or summarizing a rather long suggestion ("summary of extended text").

Example B shows a single prompt input in a text field, that is a zero-shot prompting interface as in Wordcraft [67], and Script&Shift [49]. Similar to a conversational UI, these text fields offer users the opportunity to reach every point within the space of AI functions, but with only one try (assuming a basic version of this UI element without additional features). This free prompting offers a maximum of functional flexibility to the user and users have to carefully describe the intended function because this shapes the targeted subspace. Findings on the prompting behavior with our second prototype show that users specified parameters in their prompts to shape the target function ("Extend: write about *(some meals)*, *(that can be made with tomatoes)*", and "Reduce this text *(by 50 words)*"; we put the parameters in parentheses/italic).

Example C shows a toolbar UI, similar to AI toolbars in industry solutions such as Grammarly. This type of UI constrains the functional flexibility more than the examples before. Such toolbars offer pre-determined tools that capture commonly desired functions informed through user research. They thus provide a set of AI functions that the user can execute without entering a text prompt. Since these functions are pre-determined (e.g. via prompts hardcoded in the backend), they reach only certain functional subspaces, as defined by system builders. Based on the interactions with our second prototype, we found users to intuitively apply these AI functions, and chain them in various ways to explore the pre-determined subspaces (pre-determined AI functions accounted for 64.42% of used functions, the others were used via free prompting).

8.3 Functional flexibility as a thinking tool

We conclude with conceptual reflections and application examples. Conceptually, our functional space is different from design spaces, which describe the options for (UI) designs, for example, structuring design requirements for co-creation [32] or connecting HCI demands with model capabilities [40]. In contrast, our conceptual lens describes how different UIs make AI functions accessible in different ways. This accounts for more than prompt-based interfaces [53]: As a key conceptual contribution, we offer a perspective for analyzing and comparing UIs that provide AI capabilities and that are as different as CUIs, traditional GUIs, and text-based prompting UIs.

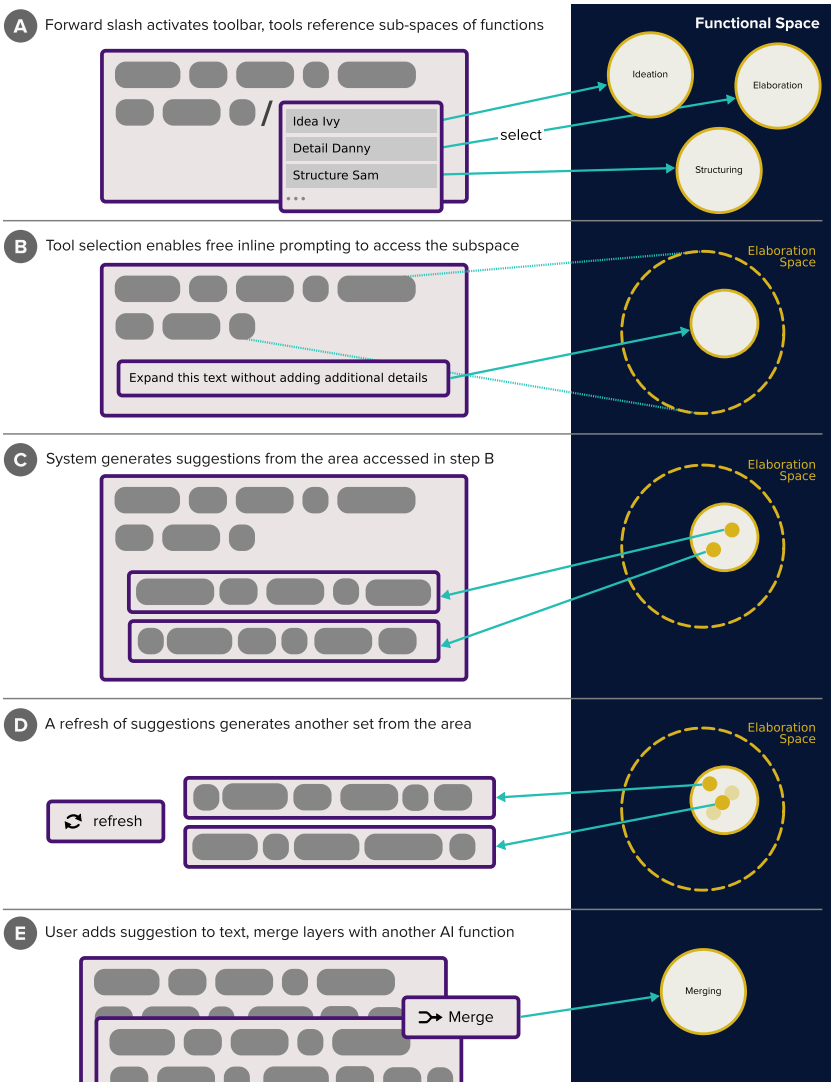


Fig. 13. Illustration of how Script&Shift [49] enabled accessing the functional space of the LLM. The illustration shows how (A) typing a forward slash triggers displaying an AI toolset. Users can select one of these functions. (B) Selecting a tool scopes a subspace, an inline free prompting text field allows users exploring this subspace. (C) The system generates a pair of suggestions. (D) Refreshing suggestions generates a new set. (E) Users can add the suggestion to the text. If multiple layers exist, they can be combined through a pre-determined merge function.

It is possible to analyze further designs with this lens. For example, we discuss the interface from the application Script&Shift by Siddiqui et al. [49] through the lens of our construct by answering analytical, critical, and constructive questions from Table 9.

In Figure 13, we visualized how writing with the tool Script&Shift enabled accessing the functional space of an LLM. We focus on their features “text elaboration” and “merging layers”. (A) When people type a forward slash “/”, an inline

toolbar appears, offering six "Writer's Friends", such as Idea Ivy for ideation, and Detail Danny for elaborations. These Writer's Friends refer to subspaces of functions in the functional space. Users can select one of these Friends to scope a subspace. The tools confine the functional space of the LLM for the user. (B) An input field for free prompting appears below the paragraph, and gives the user maximum flexibility to navigate the subspace referenced through the previous selected Writer's Friend. The user is free to specify further instructions to the model, for example "expand this text without adding addition details" (cf. [49]). (C) By conditioning the LLM's generation on the user's entered prompt, the system generates two suggestions. The user can choose one of the two suggestions for including it into the original text. (D) A re-generation of the suggestions triggers the system to serve two other suggestions from the same subspace, as in the previous step. (E) With Scrip&Shift, it is possible to write in multiple layers. Their concept of layers is similar to unconnected pages that can be interacted with on an infinite canvas space. Some Writer's Friends generate such pages as an output, for example, Structure Sam generates two differently organized variants of the current page. Such pages can be merged into a single page by letting the LLM combine them.

Dissecting the UI and the interaction flow in this way shows that the application involves a combination of an AI toolbar and an inline text input field for free prompting, similarly illustrated in example B in our **??**. Selecting a tool (Writer's Friend) from the AI toolset (list of Friends) scopes a subspace (low flexibility) and the free prompting UI element gives back maximum flexibility to navigate this subspace. Dragging and dropping one page onto another, scopes again a pre-determined subspace (low flexibility) to let the LLM merge the two pages. This answers the analytical question from Table 9 "What UI elements are available with which degree of functional flexibility?". The AI toolbar offers six pre-determined functions, which guide the user with decreased functional flexibility. However, some users might want to specify their own functions for maximum flexibility, i.e. free prompting without selecting one of the predefined Writer's Friends before. These users could benefit from defining their own Writer's Friend to reach their goals, and thus more freely explore the functional space. This answers the critical question "Can users reach/identify their goals with the degree of functional flexibility offered by the system?". Adding another trigger, such as typing a square "#" could open an inline text input field for free-prompting own functions. These functions could be named and added to the toolbar, through checking a box next to the input field and adding a name. This way, users could add repeatedly used AI functions as further Writer's Friends. This free-prompting of functions (maximum flexibility) and making them available as reusable Writer's Friends (confined flexibility) would allow for dynamic switches of functional flexibility. This answers the constructive question "Which UI elements could be added to a design to increase/decrease functional flexibility?".

Reflecting on Script&Shift through the lens of Functional Flexibility enabled us to dissect the interaction analytically, as well as constructively criticizing the application. This demonstrates the value of Functional Flexibility as a thinking tool for building human-AI interaction.

More generally, we hope that our theoretical construct serves as a useful thinking tool for researchers, designers, and engineers to analyze and reflect on how specific UI implementations support users in locating desired functions within the functional space provided by LLMs. Crucially, this provides a meaningful level of abstraction to comparatively discuss *both* conversational and non-conversational UI choices. We deem this relevant and impactful in the current context, where many AI features in practice seem to predominantly consider conversational interaction.

9 Discussion

9.1 Differentiating functional flexibility from other concepts in Human-AI interaction

How does functional flexibility relate to other concepts relevant to Human-AI interaction, such as affordance, intent formulation, task decomposition, and mental models? We discuss this here:

Affordance defines how UIs make their function apparent [42]. Thus, UI elements should visibly signal the amount of functional flexibility they are offering to the user, for example the size of a text input field may indicate greater functional flexibility than a single line input field, or even a button.

Formulating intent in free text (prompting) is demanding for users [68] but may offer the possibility to scope an intent with a higher degree of functional flexibility. However, decreased functional flexibility may allow users to express intentions more quickly if they are offered in the (limited set of the) UI as a tool.

Such tools can also support *decomposing a task* bottom up (e.g. first translating and then summarizing a text), as their presence in the UI already reveals these available AI functions as possible "steps". Higher functional flexibility instead allows for flexibly decomposing tasks (top-down) but imposes metacognitive demands on the user to do so [56].

Interacting with different degrees of functional flexibility may also influence the *mental model*, that is, how the user thinks that the system works. A UI with high functional flexibility (e.g. prompt box, chatbot) may be more likely to be thought of as an agent, assistant, or partner – with users leaving the mental model at that high level of abstraction. Conversely, UIs with lower functional flexibility (e.g. a toolbar) may result in more specific mental models because each individual UI element (e.g. button in the toolbar) can be more clearly linked to a more specific expected function and output.

In summary, this short discussion in light of other concepts illustrates how functional flexibility can extend and refine our conceptual understanding of human-AI interaction.

9.2 Conversation is not all you need: "regression to commands" in repeated prompting

We offered a conversational UI in study 1, based on the rich task delegations elicited in the survey. However, participants did not actually delegate tasks in this rich way. Instead, they went with command-like instructions and optimized their prompts down to a minimum, for example, writing "extend", or even more extreme, "extd", for delegating the AI to extend the marked text.

We expect similar effects to show up in various prompting-based interaction contexts, not limited to writing. We propose to refer to this effect as *"regression to commands"*. From the user perspective, this is optimizing or satisficing [50], considering the trade-off of input effort vs result quality. Complementary, from a researcher perspective, it is something to be aware of in methodological considerations: We might elicit diverse prompts in one sample (e.g. our survey) because people can think of elaborate prompts. However, repeatedly eliciting prompts in an actual workflow reveals the (possibly much more narrow) "point" that is good enough to access that model functionality in interaction.

In contrast to the first study, in study 2, we offered direct tools for the repeated tasks, plus an optional free prompt input. In this setup, participants used the free prompt input selectively (35.58% of AI triggers), but when they used it, they indeed entered more elaborate prompts, as reflected in Figure 11.

Critically and constructively, this has strong implications for human-AI interaction research and design: Current AI is often integrated into products as a chatbot, heavily biased by ChatGPT's success, but this ignores a fundamental need in people's actual workflows. These often involve *repeated* subtasks, which in turn require repeated access to specific functionality from the LLM's space of functions. As our conceptual lens highlights (Figure 12), CUIs are great to navigate

the space towards a target function. However, CUIs are not ideal for repeatedly accessing the same target function, because this demands reentering prompts again and again. Instead, this repeated use is much better addressed by a toolbar and, more generally, by direct manipulation interfaces (e.g. [38]) that allows users to access LLM functionality without text entry.

9.3 The functional flexibility funnel: turning users' free prompts into future tools

In another constructive application of functional flexibility, we propose to consider how the *parallel* use of UI elements that provide different degrees of functional flexibility, such as free prompt inputs and toolbar-like UIs, can serve as a user research method to elicit functional user needs within a system even after its deployment [1, 62].

With UI elements that provide specific AI functions, for example, the summary tool in our AI toolbar, it may happen that there are mismatches between the provided function and the user's expectation. This was the case, for example, when the summaries were too short or did not reflect important aspects of the original content. As a mitigation strategy, participants used the UI element for free prompt input to declare their own functions, such as "summarize *every section*" and "summarize text: *4-5 sentences*". In these examples, they parameterized the scope of the text document and the text length. In other prompts, they specified parameters for the tone and topic, such as "rewrite to feel more *natural*", "Write something about the *taste of tomatoes*". In doing so, they provided us with data about the actual function they would have liked to access.

Thus, a key insight here is that these parameters could be directly integrated into the UI. For example, a toolbar UI for extending or summarizing text could display a slider to adjust the text length. *DynaVis* [59] proposed dynamically generated widgets to realize a similar idea for prompting-based interaction with information visualizations. Beyond UI generation, the users' prompts could also be used to inform software updates. For example, system builders could use them to identify and inform *fundamentally new* AI functions to be provided (as tools) in an update.

In this way, the application's UI could provide more and more refined tools for direct access to AI functions over its lifecycle. From an initially flexible space of functions accessed via prompt input, users guide the development towards a practically relevant toolset, accessed without prompting. In turn, (future) users are guided in their use and prompting needs by the already available tools. Metaphorically, we refer to this refinement of prompting needs as the *functional flexibility funnel*. Note that this does not mean that flexibility is increasingly restricted – the UI might always offer a free prompt option.

9.4 The design of user interfaces shapes how users access functional flexibility

Offering a maximum of functional flexibility to users might be tempting for designers and developers. For example, at first glance, a simple input field for free prompting seems to replace the need to design any other UI elements. We currently see this strategy in many products, such as ChatGPT or Microsoft Copilot – their UIs are mainly built on and marketed as a promise of the flexibility of an open text box. However, applying our lens of functional flexibility critically, this *shifts the responsibility for locating AI functions* completely to the user.

Instead, designers and engineers could identify and implement low-flexibility UI elements as *starting points* for users to access (some) AI functions more easily. Such pre-determined functions (e.g. offered as buttons) lower the barrier for using AI functions since users do not have to locate these functions in the space by themselves.

Indeed, study 2 revealed that participants were initially clueless about prompting and asked for examples (e.g. "I wish for a list of examples, I was not sure if I was free about text input"). However, these participants referred to

pre-determined AI functions for their first prompts: They input short commands that could possibly extend the toolbar, such as “itemize”, or specifically extended existing functions, such as “summarize 2-3 sentences less”.

Thus, pre-determined functions and their related UI elements can also serve as examples that set perceivable “landmarks” in the functional space: They reveal a subset of what is possible and thus *shape user expectations about the whole space*.

Overall, our proposed perspective enables researchers, designers, and engineers to better assess and compare the various possibilities of UIs to shape functional flexibility. In particular, we expect that making functional flexibility an explicit consideration in the design of interactive AI systems helps reveal design opportunities beyond (or in combination with) conversational UIs and simple prompt boxes.

9.5 Further findings: design explorations in time and space

Here we discuss further implications of our empirical work.

9.5.1 Time: unlocking parallel human-AI workflows by designing with the timing of AI involvement. Designing timing of AI involvement in writing is underexplored. A recent survey [29] lists the design dimension of “initiative” (system vs user) but not delays or further temporal aspects. This indicates that AI-induced delays are currently seen as an (unfortunate) byproduct of a system implementation but not a material to actively design *with*. Our findings motivate a change in perspective. Adding a few seconds of delay as AI “typing time” in our conversational UI (study 1) already led to people working on other tasks in parallel, such as writing, reading, thinking, and delegating further tasks to the AI. The AI tools (study 2) without such delays were also used in parallel workflows (Section 7.2.3) by keeping (multiple) AI comment cards open while working on further tasks. These findings suggest that timing in human-AI workflows deserves more attention and should be explored further. For example, more extensive AI response delays could be interesting for writing projects with multiple collaborators (e.g. to let multiple humans chime in before the AI does) or postponed tasks.

9.5.2 Space: using annotations to enable “put that there”. Current writing assistants are often integrated as a sidebar [67], decoupled from the scroll state of the page UI. This makes it hard to refer to a “there” (cf. [10]) in the document. Based on our findings, we argue that “annotation” is a better integration concept. Annotations allow users to point the AI to specific context when expressing their intention (in a prompt) and thus help them navigate the functional space.

We reused collaboration UIs (comments) for this, which participants appreciated for parallelism (“I would find any other placement frustrating if this was blocking me”), keeping things in sight (“Feels good. Placed right and does not occlude text”) and ready to hand (“Makes sense, it was good that the cards were kept there and you could copy things from them”).

Furthermore, text selection serves as a “scope” parameter, as we discovered in our survey. Tasks like summarizing, translating, or extending text can then be instructed very efficiently (e.g. simply writing “extend”). Annotating the scope thus removes the burden of specifying it in the written prompt. Finally, annotations allow for designs in which the AI can act beyond the user’s viewport. This out-of-focus AI editing should be explored more as working on different document parts is essential in human collaboration [61].

9.6 Limitations

We followed a mixed-methods approach and observed participants’ screens via video calls. This approach offered insights we otherwise would have missed. For example, observations revealed how participants chained AI functions.

People also shared their thoughts *during* writing, which not only provides a glimpse at their thinking processes but also helps us to interpret the findings from interaction logging (cf. [8]). Overall, this investment comes with typical tradeoffs of qualitative research, such as favoring detailed observation over a larger sample and risking that observations might influence people's behavior.

We do not claim to have achieved an "optimal" implementation of the AI features and this would be a moving target anyway, since model development advances quickly. "Perfect" text generation was not required for our research here, as long as AI-generated text could be realistically included in writing workflows. This bar was met for the tasks in our studies, based on our own assessments and people's perception.

10 Conclusion

We investigated interaction with text models in multiple studies over four years. Within this period falls the moment when writing with conversational AI became widely popular. Our work includes a survey on text editor usage and task delegation to AI without the "legacy bias" of ChatGPT and two user studies: one with a conversational UI and another with a toolbar UI, offering pre-determined AI functions alongside custom prompt input. When interacting with the conversational UI, people used short command-like instructions for task delegation, in contrast to the more elaborate prompts elicited in our survey. With the toolbar UI providing *both* pre-specified AI tools and flexible prompting, people used and combined the tools for specific tasks, while they invested into free prompting to realize custom functions.

Moreover, while AI tools were designed to execute on request, agents were designed with a delay. This timing provides opportunities for the design of AI involvement, shaping parallelism in human-AI workflows.

Synthesizing our findings with contributions from related work, we proposed the new theoretical construct and perspective of *functional space* and *functional flexibility*. We reflected on our prototypes and related work in light of functional flexibility, considering analytical, critical, and constructive perspectives. This reveals and makes explicit a new key mechanism by which "interaction with LLMs" can be construed in HCI: accessing a desired target function through suitable UIs. In this light, UIs serve the key role of granting users access to the functional space of LLMs and their design shapes how users can do so, leading to different degrees of functional flexibility. With our work, we thus contribute to function-oriented design of human-AI interaction with generative AI.

Although it is tempting to offer a maximum of flexibility to users by letting them decide on relevant functionality via prompting themselves, our investigation shows that effectively integrating AI into UIs requires actively thinking about, and designing for, providing features with the right amount(s) of functional flexibility.

References

[1] Saleema Amershi, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, Eric Horvitz, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, and Paul N. Bennett. 2019. Guidelines for Human-AI Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. ACM Press, New York, NY, USA, 1–13. doi:10.1145/3290605.3300233

[2] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena L. Glassman. 2024. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–18. doi:10.1145/3613904.3642016

[3] Kenneth C. Arnold, Krysta Chauncey, and Krzysztof Z. Gajos. 2020. Predictive text encourages predictable writing. In *Proceedings of the 25th International Conference on Intelligent User Interfaces (IUI '20)*. Association for Computing Machinery, Cagliari, Italy, 128–138. doi:10.1145/3377325.3377523

[4] Aaron Bangor, Philip T. Kortum, and James T. Miller. 2008. An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction* 24, 6 (July 2008), 574–594. doi:10.1080/10447310802205776

[5] Michel Beaudouin-Lafon. 2000. Instrumental interaction: an interaction model for designing post-WIMP user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, The Hague The Netherlands, 446–453. doi:10.1145/332040.332473

[6] Michel Beaudouin-Lafon, Susanne Bødker, and Wendy E. Mackay. 2021. Generative Theories of Interaction. *ACM Transactions on Computer-Human Interaction* 28, 6 (Dec. 2021), 1–54. doi:10.1145/3468505

[7] Dan Bennett, Oussama Metatla, Anne Roudaut, and Elisa Mekler. 2023. How does HCI Understand Human Autonomy and Agency? doi:10.1145/3544548.3580651 arXiv:2301.12490 [cs].

[8] Advait Bhat, Saaket Agashe, Parth Oberoi, Niharika Mohile, Ravi Jangir, and Anirudha Joshi. 2023. Interacting with Next-Phrase Suggestions: How Suggestion Systems Aid and Influence the Cognitive Processes of Writing. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. ACM, Sydney NSW Australia, 436–452. doi:10.1145/3581641.3584060

[9] Jeremy Birnholtz, Stephanie Steinhardt, and Antonella Pavese. 2013. Write here, write now! an experimental study of group maintenance in collaborative writing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. Association for Computing Machinery, Paris, France, 961–970. doi:10.1145/2470654.2466123

[10] Richard A. Bolt. 1980. "Put-that-there": Voice and gesture at the graphics interface. *ACM SIGGRAPH Computer Graphics* 14, 3 (July 1980), 262–270. doi:10.1145/965105.807503

[11] John Brooke. 1996. SUS: A 'Quick and Dirty' Usability Scale. In *Usability Evaluation In Industry* (1st ed. ed.). CRC Press, 6. https://www.taylorfrancis.com/chapters/edit/10.1201/9781498710411-35/sus-quick-dirty-usability-scale-john-brooke?context=ubx&refId=9a8e38ab-850d-4b18-b3c6-59b619a7ff61

[12] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]* (July 2020). http://arxiv.org/abs/2005.14165 arXiv: 2005.14165.

[13] Daniel Buschek, Martin Zürn, and Malin Eiband. 2021. The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers. *arXiv:2101.09157 [cs]* (Jan. 2021). doi:10.1145/3411764.3445372 arXiv: 2101.09157.

[14] Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative Writing with a Machine in the Loop: Case Studies on Slogans and Stories. In *23rd International Conference on Intelligent User Interfaces*. ACM, Tokyo Japan, 329–340. doi:10.1145/3172944.3172983

[15] Hai Dang, Florian Lehmann, Sven Goller, and Daniel Buschek. 2023. Choice Over Control: How Users Write with Large Language Models using Diegetic and Non-Diegetic Prompting. (2023).

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423

[17] Paramveer S. Dhillon, Somayeh Molaei, Jiaqi Li, Maximilian Golub, Shaochun Zheng, and Lionel Peter Robert. 2024. Shaping Human-AI Collaboration: Varied Scaffolding Levels in Co-writing with Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–18. doi:10.1145/3613904.3642134

[18] Gijsbert Erkens, Jos Jaspers, Maaike Prangsma, and Gellof Kanselaar. 2005. Coordination processes in computer supported collaborative writing. *Computers in Human Behavior* 21, 3 (May 2005), 463–486. doi:10.1016/j.chb.2004.10.038

[19] Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence Compression by Deletion with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 360–368. doi:10.18653/v1/D15-1042

[20] Liye Fu, Benjamin Newman, Maurice Jakesch, and Sarah Kreps. 2023. Comparing Sentence-Level Suggestions to Message-Level Suggestions in AI-Mediated Communication. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–13. doi:10.1145/3544548.3581351

[21] Sebastian Gehrmann, Hendrik Strobelt, Robert Kruger, Hanspeter Pfister, and Alexander M. Rush. 2019. Visual Interaction with Deep Learning Models through Collaborative Semantic Inference. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2019), 884–894. doi:10.1109/TVCG.2019.2934595

[22] Han L Han, Miguel A Renom, Wendy E Mackay, and Michel Beaudouin-Lafon. 2020. Textlets: Supporting Constraints and Consistency in Text Documents. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems - CHI '20*. ACM, New York, NY, USA, 13. doi:10.0.4.121/3313831.3376804

[23] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '99)*. ACM Press, New York, NY, USA, 159–166. doi:10.1145/302979.303030

[24] Maurice Jakesch, Advait Bhat, Daniel Buschek, Lior Zalmanson, and Mor Naaman. 2023. Co-Writing with Opinionated Language Models Affects Users' Views. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 111, 15 pages. doi:10.1145/3544548.3581196

[25] Ellen Jiang, Kristen Olson, Edwin Toh, Alejandra Molina, Aaron Donsbach, Michael Terry, and Carrie J Cai. 2022. PromptMaker: Prompt-based Prototyping with Large Language Models. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, Article 35, 8 pages. doi:10.1145/3491101.3503564

[26] Tae Soo Kim, Minsuk Chang, Yoonjoo Lee, and Juho Kim. 2023. Cells, Generators, and Lenses: Design Framework for Object-Oriented Interaction with Large Language Models. (2023).

[27] Janin Koch, Andrés Lucero, Lena Hegemann, and Antti Oulasvirta. 2019. May AI?: Design Ideation with Cooperative Contextual Bandits. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. ACM Press, New York, NY, USA, 1–12. doi:10.1145/3290605.3300863

[28] Janin Koch, Nicolas Taffin, Michel Beaudouin-Lafon, Markku Laine, Andrés Lucero, and Wendy E. Mackay. 2020. ImageSense: An Intelligent Collaborative Ideation Tool to Support Diverse Human-Computer Partnerships. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (May 2020), 1–27. doi:10.1145/3392850

[29] Mina Lee, Katy Ilonka Gero, John Joon Young Chung, Simon Buckingham Shum, Vipul Raheja, Hua Shen, Subhashini Venugopalan, Thiemo Wambsganss, David Zhou, Emad A. Alghamdi, Tal August, Avinash Bhat, Madiha Zahrah Choksi, Senjuti Dutta, Jin L.C. Guo, Md Naimul Hoque, Yewon Kim, Simon Knight, Seyed Parsa Neshaei, Antonette Shibani, Disha Shrivastava, Lila Shroff, Agnia Sergeyuk, Jessi Stark, Sarah Sterman, Sitong Wang, Antoine Bosselut, Daniel Buschek, Joseph Chee Chang, Sherol Chen, Max Kreminski, Joonsuk Park, Roy Pea, Eugenia Ha Rim Rho, Zejiang Shen, and Pao Siangliulue. 2024. A Design Space for Intelligent and Interactive Writing Assistants. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 1054, 35 pages. doi:10.1145/3613904.3642697

[30] Mina Lee, Percy Liang, and Qian Yang. 2022. CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–19. doi:10.1145/3491102.3502030

[31] Florian Lehmann, Niklas Markert, Hai Dang, and Daniel Buschek. 2022. Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship. In *Mensch und Computer 2022*. ACM, Darmstadt Germany, 192–208. doi:10.1145/3543758.3543947

[32] Zhiyu Lin, Upol Ehsan, Rohan Agarwal, Samihan Dani, Vidushi Vashishth, and Mark Riedl. 2023. Beyond Prompts: Exploring the Design Space of Mixed-Initiative Co-Creativity Systems. (2023). doi:10.48550/ARXIV.2305.07465 Publisher: arXiv Version Number: 1.

[33] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–31. doi:10.1145/3544548.3580817

[34] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. http://arxiv.org/abs/1907.11692 arXiv:1907.11692 [cs].

[35] Zhicong Lu, Mingming Fan, Yun Wang, Jian Zhao, Michelle Annett, and Daniel Wigdor. 2019. InkPlanner: Supporting Prewriting via Intelligent Visual Diagramming. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (Jan. 2019), 277–287. doi:10.1109/TVCG.2018.2864887

[36] Wendy E. Mackay and Michel Beaudouin-Lafon. 2025. Interaction Substrates: Combining Power and Simplicity in Interactive Systems. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–16. doi:10.1145/3706598.3714006

[37] Neil Maiden, Konstantinos Zachos, Amanda Brown, George Brock, Lars Nyre, Aleksander Nygård Tonheim, Dimitris Apsotolou, and Jeremy Evans. 2018. Making the News: Digital Creativity Support for Journalists. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, NY, USA, 1–11. doi:10.1145/3173574.3174049

[38] Damien Masson, Sylvain Malacria, Géry Casiez, and Daniel Vogel. 2024. DirectGPT: A Direct Manipulation Interface to Interact with Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 975, 16 pages. doi:10.1145/3613904.3642462

[39] Meredith Ringel Morris. 2024. Prompting Considered Harmful. *Commun. ACM* 67, 12 (Nov. 2024), 28–30. doi:10.1145/3673861

[40] Meredith Ringel Morris, Carrie J. Cai, Jess Holbrook, Chinmay Kulkarni, and Michael Terry. 2023. The Design Space of Generative Models. (2023). doi:10.48550/ARXIV.2304.10547 Publisher: arXiv Version Number: 1.

[41] Sheshera Mysore, Debarati Das, Hancheng Cao, and Bahareh Sarrafzadeh. 2025. Prototypical Human-AI Collaboration Behaviors from LLM-Assisted Writing in the Wild. doi:10.48550/arXiv.2505.16023 arXiv:2505.16023 [cs].

[42] Donald A. Norman. 2002. *The Design of Everyday Things*. Basic Books, Inc., USA.

[43] QuillBot. 2025. QuillBot. https://quillbot.com

[44] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019), 24. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. http://arxiv.org/abs/1910.10683 arXiv:1910.10683 [cs, stat].

[46] Leon Reicherts, Nima Zargham, Michael Bonfert, Yvonne Rogers, and Rainer Malaka. 2021. May I Interrupt? Diverging Opinions on Proactive Smart Speakers. In *CUI 2021 - 3rd Conference on Conversational User Interfaces*. ACM, Bilbao (online) Spain, 1–10. doi:10.1145/3469595.3469629

[47] Laria Reynolds and Kyle McDonell. 2021. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–7. doi:10.1145/3411763.3451760

[48] Nathalie Riche, Anna Offenwanger, Frederic Gmeiner, David Brown, Hugo Romat, Michel Pahud, Nicolai Marquardt, Kori Inkpen, and Ken Hinckley. 2025. AI-Instruments: Embodying Prompts as Instruments to Abstract & Reflect Graphical Interface Commands as General-Purpose Tools. doi:10.1145/3706598.3714259 arXiv:2502.18736 [cs].

[49] Momin Siddiqui, Roy Pea, and Hari Subramonyam. 2025. Script&Shift: A Layered Interface Paradigm for Integrating Content Development and Rhetorical Strategy with LLM Writing Assistants. doi:10.48550/arXiv.2502.10638 arXiv:2502.10638 [cs].

[50] Herbert A. Simon. 1996. *The sciences of the artificial (3rd ed.).* MIT Press, Cambridge, MA, USA.

[51] Nikhil Singh, Guillermo Bernal, Daria Savchenko, and Elena L. Glassman. 2022. Where to Hide a Stolen Elephant: Leaps in Creative Writing with Multimodal Machine Intelligence. *ACM Transactions on Computer-Human Interaction* (Feb. 2022), 3511599. doi:10.1145/3511599

[52] Sarah Sterman, Evey Huang, Vivian Liu, and Eric Paulos. 2020. Interacting with Literary Style through Computational Tools. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–12. doi:10.1145/3313831.3376730

[53] Hari Subramonyam, Roy Pea, Christopher Pondoc, Maneesh Agrawala, and Colleen Seifert. 2024. Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–19. doi:10.1145/3613904.3642754

[54] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–26. doi:10.1145/3613904.3642400

[55] Ben Swanson, Kory Mathewson, Ben Pietrzak, Sherol Chen, and Monica Dinalescu. 2021. Story Centaur: Large Language Model Few Shot Learning as a Creative Writing Tool. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Online, 244–256. doi:10.18653/v1/2021.eacl-demos.29

[56] Lev Tankelevitch, Viktor Kewenig, Auste Simkute, Ava Elizabeth Scott, Advait Sarkar, Abigail Sellen, and Sean Rintel. 2024. The Metacognitive Demands and Opportunities of Generative AI. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–24. doi:10.1145/3613904.3642902

[57] Maartje ter Hoeve, Robert Sim, Elnaz Nouri, Adam Fourney, Maarten de Rijke, and Ryen W. White. 2020. Conversations with Documents: An Exploration of Document-Centered Assistance. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval*. ACM, Vancouver BC Canada, 43–52. doi:10.1145/3343413.3377971

[58] Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – Building open translation services for the World. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, André Martins, Helena Moniz, Sara Fumega, Bruno Martins, Fernando Batista, Luisa Coheur, Carla Parra, Isabel Trancoso, Marco Turchi, Arianna Bisazza, Joss Moorkens, Ana Guerberof, Mary Nurminen, Lena Marg, and Mikel L. Forcada (Eds.). European Association for Machine Translation, Lisboa, Portugal, 479–480. https://aclanthology.org/2020.eamt-1.61

[59] Priyan Vaithilingam, Elena L. Glassman, Jeevana Priya Inala, and Chenglong Wang. 2024. DynaVis: Dynamically Synthesized UI Widgets for Visualization Editing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–17. doi:10.1145/3613904.3642639

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Long Beach, California, USA, 6000–6010.

[61] Dakuo Wang, Haodan Tan, and Tun Lu. 2017. Why Users Do Not Want to Write Together When They Are Writing Together: Users' Rationales for Today's Collaborative Writing Practices. *Proceedings of the ACM on Human-Computer Interaction* 1, CSCW (Dec. 2017), 1–18. doi:10.1145/3134742

[62] Justin D. Weisz, Jessica He, Michael Muller, Gabriela Hoefer, Rachel Miles, and Werner Geyer. 2024. Design Principles for Generative AI Applications. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–22. doi:10.1145/3613904.3642466

[63] Geraint A. Wiggins. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19, 7 (Nov. 2006), 449–458. doi:10.1016/j.knosys.2006.04.009

[64] Geraint A. Wiggins. 2019. A Framework for Description, Analysis and Comparison of Creative Systems. In *Computational Creativity*, Tony Veale and F. Amílcar Cardoso (Eds.). Springer International Publishing, Cham, 21–47. doi:10.1007/978-3-319-43610-4_2 Series Title: Computational Synthesis and Creative Systems.

[65] Tongshuang Wu, Michael Terry, and Carrie J. Cai. 2021. AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts. *arXiv:2110.01691 [cs]* (Oct. 2021). http://arxiv.org/abs/2110.01691 arXiv: 2110.01691.

[66] Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T. Iqbal, and Jaime Teevan. 2019. Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. ACM Press, New York, NY, USA, 1–12. doi:10.1145/3290605.3300415

[67] Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: Story Writing With Large Language Models. In *27th International Conference on Intelligent User Interfaces*. ACM, Helsinki Finland, 841–852. doi:10.1145/3490099.3511105

[68] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–21. doi:10.1145/3544548.3581388

[69] Wenjie Zhong, Jason Naradowsky, Hiroya Takamura, Ichiro Kobayashi, and Yusuke Miyao. 2023. Fiction-Writing Mode: An Effective Control for Human-Machine Collaborative Writing. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Dubrovnik, Croatia, 1752–1765. doi:10.18653/v1/2023.eacl-main.128

A Appendix

A.1 Questionnaires

The complete questionnaire of our survey can be found in our OSF repository[8] alongside the other questionnaires in our studies. Since the latter were rather short questionnaires, we also put them into this appendix.

A.2 Task Briefing

A.2.1 Opener study 1: Here, you will use an online text editor with AI skills.

The AI skills support you with summarizing, extending, and translating text within the editor. By commenting parts of the text with to-dos an AI author will take over these tasks. See the screenshots below for a short demonstration.

Video: Stills of the video can be seen in Figure 14.

A.2.2 Opener study 2: Here, you will use an online text editor with an AI toolset.

The AI toolset supports you with summarizing, extending, and translating text within the editor. Furthermore, you can specify an AI action on your own using the prompt tool. By selecting parts of the text and using the AI tool from the floating toolbar, an AI action will be applied to your text. See the video below for a short demonstration.

Video: Stills of the video can be seen in Figure 15.

A.2.3 Remaining task briefing: You will have to write an informal blog post for a gardening blog. The main topic is tomato plants. In the same document, you will have to write a short summary about your blog post for impatient readers, known as "wrap up" or "too long did not read".

Information on the task:

Task: Write a blog post and a summary of that for a gardening blog.

Topic: Tomato plants.

Language: English.

Text length: The blog post must have a length between 300 - 350 words. Additionally, your summary should be concise with a length of three to five sentences maximum.

Quality: The quality of the text should be moderate/reasonable. It must not be perfect, yet you should avoid making a lot of mistakes. Try to be efficient and effective at the same time! The task does not assess your writing skills.

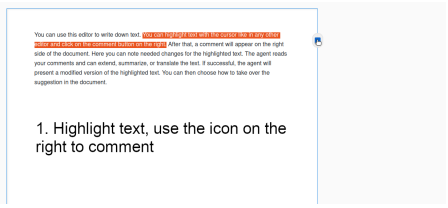
Time: You should try to finish the task within 20 - 30 minutes.

Keywords: The blog post should include the keywords as follows: Tomato, origin, summer, water, balcony, growing, fruit, taste.

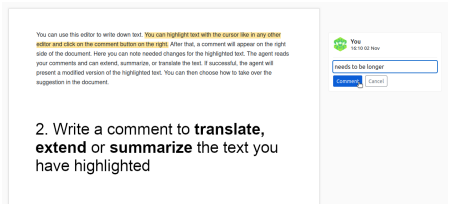
Resources: Your are allowed to use the following resources for writing your blog post:

- https://de.wikipedia.org/wiki/Tomate
- https://www.mein-schoener-garten.de/pflanzen/gemuse/tomaten
- https://www.gartentipps.com/tomaten-auf-dem-balkon-ziehen-wertvolle-tipps-zum-anbau.html
- https://www.plantura.garden/gartentipps/gemuseratgeber/tomaten-anbau-auf-terrasse-und-balkon
- https://www.poetschke.de/beratung/tomate-ratgeber/
- https://www.native-plants.de/blog/tomatenpflanzen-selbst-ziehen/

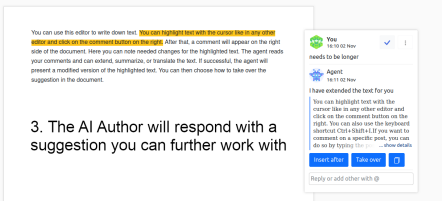
[8] https://osf.io/qh46n/?view_only=e8b7ab117cea48118d192e5eac7048bf



(a) Step one

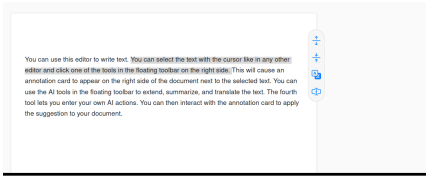


(b) Step two



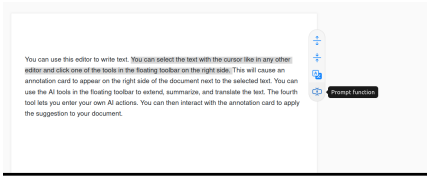
(c) Step three

Fig. 14. Stills from the video we presented to participants within the task briefing of study 1. The stills explain in three steps how a writing task can be delegated to AI.



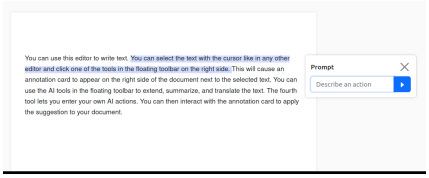
Start by **selecting a part of the text** with the cursor, then use the **floating toolbar on the right**.

(a)



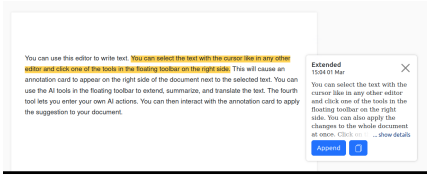
and **prompt custom AI actions**

(b)



by describing the action on your own.

(c)



Finally, the system will present a suggestion and several options to apply it to your document.

(d)

Fig. 15. Stills from the task briefing video we presented to participants in study two. The stills show in four steps how an AI function can be used from the toolbar UI (pre-determined functions, and free prompting of functions).

Table 10. Questions asked as part of study 1: A) system usability scale, B) system ratings, C) feedback, D) demographics.

ID	Question	Response type
A1	I think that I would like to use this system frequently	5-point Likert Rating
A2	I found the system unnecessarily complex	
A3	I thought the system was easy to use	
A4	I think that I would need the support of a technical person to be able to use this system	
A5	I found the various functions in this system were well integrated	
A6	I thought there was too much inconsistency in this system	
A7	I would imagine that most people would learn to use this system very quickly	
A8	I found the system very cumbersome to use	
A9	I felt very confident using the system	
A10	I needed to learn a lot of things before I could get going with this system	
B1	I found the interaction with the system natural	5-point Likert Rating
B2	I think that the system supported me to complete the task	
B3	I think that the system made me reach the goal faster	
B4	I found the system to look and feel similar to existing online text editors	
C1	I think that I would like to use this system frequently	Free text
C2	What was negative about the system?	Free text
C3	I am familiar with the topic "tomato plants"	5-point Likert Rating
D1	Please enter your age	Number
D2	What gender do you identify as?	Selection (Woman, Man, Non-binary, Prefer not to say, Prefer to self-describe)
D3	What is your profession / occupation / job title?	Free text
D4	How well do you speak English?	Selection (No knowledge of *[English / German]*, Speak poorly (beginner knowledge), Fairly well (intermediate knowledge), Well (advanced knowledge), Very well (proficient in English), Native speaker)
D5	How well do you speak German?	

A.3 Supplementary figures and tables

In the remaining appendix, we append all supplementary figures and tables.

Table 11. Questions asked as part of study 1: A) previous experience, B) system ratings, C) system usability scale, D) agency, E) feedback, F) demographics

ID	Question	Response type
A1	Please list all previous experiences you have in writing with generated text.	Multiple selection (none, Writing with word or sentence suggestions, Writing with auto-completion, Writing with auto-correction, Using the smart reply feature, Using ChatGPT, other)
B1	I found the interaction with the system natural	5-point Likert Rating
B2	I think that the system supported me to complete the task	
B3	I think that the system made me reach the goal faster	
B4	I found the system to look and feel similar to existing online text editors	
C1	I think that I would like to use this system frequently	5-point Likert Rating
C2	I found the system unnecessarily complex	
C3	I thought the system was easy to use	
C4	I think that I would need the support of a technical person to be able to use this system	
C5	I found the various functions in this system were well integrated	
C6	I thought there was too much inconsistency in this system	
C7	I would imagine that most people would learn to use this system very quickly	
C8	I found the system very cumbersome to use	
C9	I felt very confident using the system	
C10	I needed to learn a lot of things before I could get going with this system	
D1	It felt like I was in control of the text during the task	5-point Likert Rating
D2	I feel like I am the author of the text	
E1	I am familiar with the topic "tomato plants"	5-point Likert Rating
E2	What was negative about the system?	Free text
E3	What was positive about the system?	Free text
F1	Please enter your age	Number
F2	What gender do you identify as?	Selection (Woman, Man, Non-binary, Prefer not to say, Prefer to self-describe)
F3	What is your profession / occupation / job title?	Free text
F4	How well do you speak English?	Selection (No knowledge of *[English / German]*, Speak poorly (beginner knowledge), Fairly well (intermediate knowledge), Well (advanced knowledge), Very well (proficient in English), Native speaker)
F5	How well do you speak German?	

Table 12. Results of coded "Goals" participants had for delegating a task to AI in a writing stage. We discovered a desire for inspiration in the outlining stage, whereas participants aimed for efficiency and improving a text in the drafting and revising stages.

Writing Stage	Inspiration	Efficiency	Learning	Improve Text	Coordination
Outlining	102	159	113	49	15
Drafting	51	207	73	161	8
Revising	12	210	33	179	4
Coordinating	8	53	28	77	204

Table 13. Results of coded "Styles" participants applied for delegating a task to AI in a writing stage. Most often occurred an imperative style, followed by asking questions.

Writing Stage	Imperative	Question	Direct Input	Statement
Outlining	196	94	1	88
Drafting	245	80	0	34
Revising	250	87	0	24
Coordinating	252	83	3	14

Table 14. Results of coded "Roles" participants assigned to AI for delegating a task in a writing stage. We most often discovered participants to refer to AI as a co-author in the outlining stage. With advancing stages, they less often refer to AI as a co-author, but more often as an assistant.

Writing Stage	Co-Author	Advisor	Assistant	Editor
Outlining	143	66	150	22
Drafting	98	20	195	47
Revising	71	25	231	39
Coordinating	24	7	279	43

Table 15. Results of coded "Styles" participants applied for delegating different tasks to AI. We focus on the three AI capabilities: summarize, extend, and translate. We discovered a majority of elicited inputs to have an imperative style.

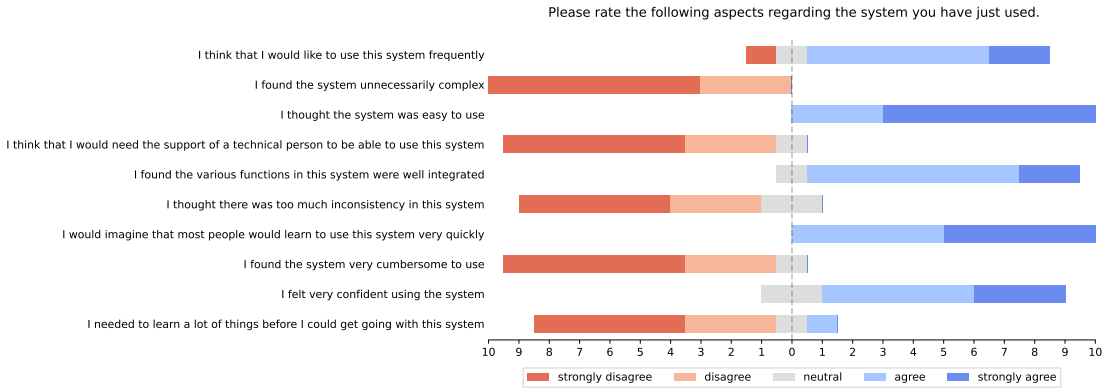
AI Capability	Imperative	Question	Direct Input	Statement
Revise	158	18	0	16
Summarize	**143**	**10**	**2**	**2**
Format	195	10	0	4
Shorten	152	17	1	4
Extend	**135**	**19**	**0**	**13**
Highlight	166	13	4	3
Cut	178	8	1	3
Complete	139	8	13	1
Draft	137	9	10	17
Inspire	155	12	3	4
Add Reference	184	3	1	5
Find Non-Text Material	163	7	2	10
Translate	**154**	**23**	**3**	**15**
Question Answering	52	115	1	21
Send	187	10	0	2
Ideation	91	36	0	14
Create Outline	144	16	0	4
Collaboration	129	20	0	5

Table 16. Results of coded "Goals" participants had for delegating different tasks to AI, with a focus on the capabilities summarize, extend, and translate. We discovered that participants aimed for efficiency, learning, and improving text when delegating a summarize task to AI. Delegating an extend task to AI they desire to gain inspiration. Translating occurs most often with the goal of efficiency.

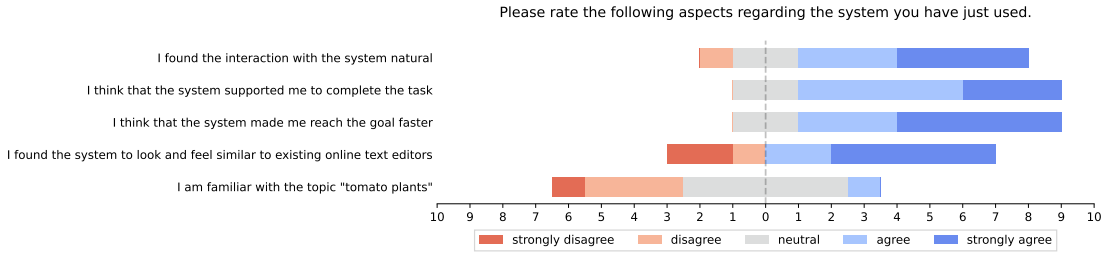
AI Capability	Inspiration	Efficiency	Learning	Improve Text	Coordination
Revise	9	42	21	136	8
Summarize	**3**	**81**	**30**	**59**	**2**
Format	0	208	1	5	0
Shorten	2	114	5	52	1
Extend	**122**	**44**	**8**	**22**	**2**
Highlight	0	176	5	27	5
Cut	0	185	1	10	0
Complete	156	2	2	4	0
Draft	142	131	4	2	0
Inspire	144	95	12	11	2
Add Reference	0	193	0	0	0
Find Non-Text Material	120	101	8	3	0
Translate	**3**	**148**	**27**	**14**	**7**
Question Answering	3	95	87	40	10
Send	0	0	0	55	199
Ideation	88	6	10	41	2
Create Outline	104	113	5	3	8
Collaboration	1	4	3	10	139

Table 17. Results of coded "Roles" in which participants saw the AI based on their message when delegating a task, with a focus on the capabilities summarize, extend, and translate (bold). We discovered that participants considered the AI most often as a co-author for all three tasks.

AI Capability	Co-Author	Advisor	Assistant	Editor
Revise	122	9	44	20
Summarize	**121**	**2**	**33**	**4**
Format	0	0	209	0
Shorten	144	0	20	9
Extend	**149**	**3**	**13**	**4**
Highlight	10	1	170	7
Cut	4	1	187	0
Complete	159	0	2	2
Draft	158	4	11	1
Inspire	153	5	7	14
Add Reference	0	0	192	0
Find Non-Text Material	19	0	161	1
Translate	**159**	**10**	**23**	**5**
Question Answering	35	9	128	18
Send	0	1	197	0
Ideation	68	24	15	38
Create Outline	141	7	12	4
Collaboration	3	2	144	6

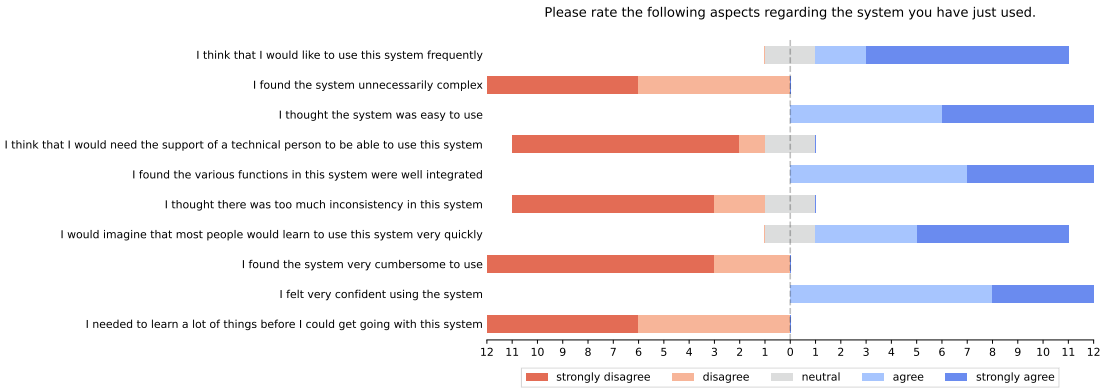


(a) The responses on the SUS questionnaire in our first user study showed a a mean score of 83.5, that is comparable to an “excellent usability” [4]

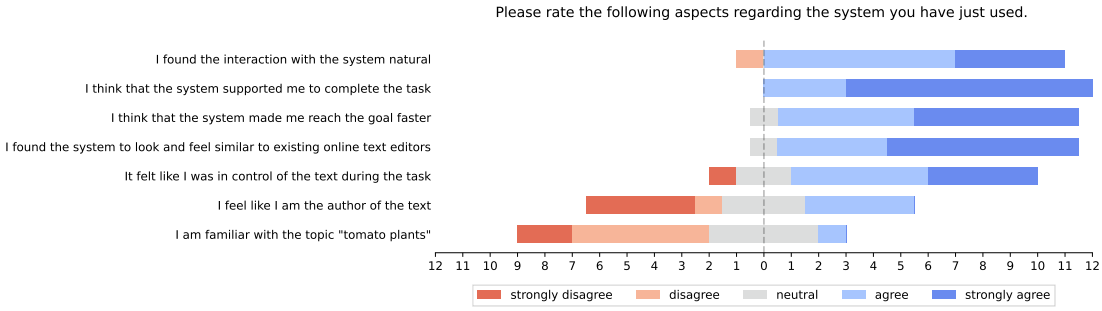


(b) Ratings on questions, asking for system usage and topic familiarity.

Fig. 16. Divergent bar charts of responses on the SUS and additional ratings in study 1.



(a) Overview of responses on the SUS questionnaire of our second user study, with a mean score of 87.29, that is comparable to an "excellent usability" [4]. The score is slightly higher than in study 1.



(b) Ratings on user perception in the seconds user study, asking for tool usage, perceived control, authorship, and topic familiarity

Fig. 17. Divergent bar charts of responses on the SUS and additional ratings in study 2.

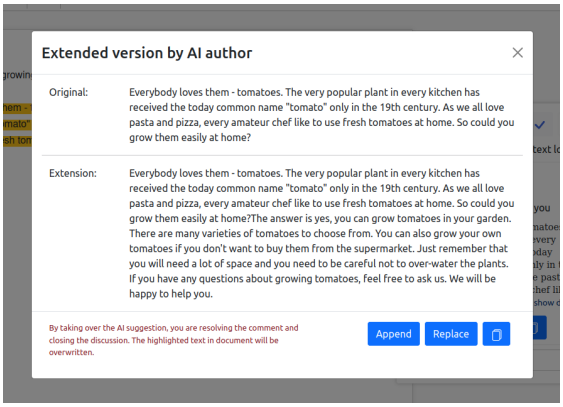


Fig. 18. Screenshot of the detail view ("diff view") that enabled users to compare their original selected text and the AI's response/suggestion.

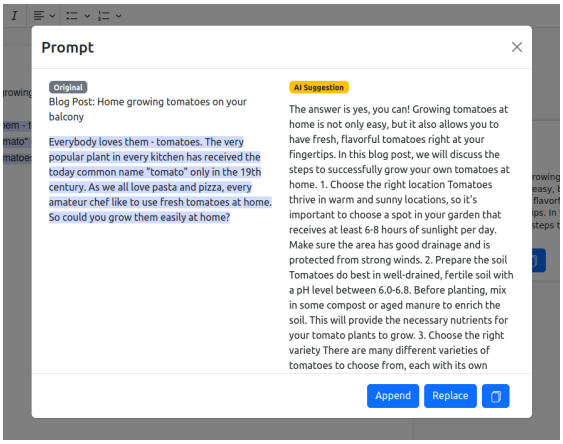


Fig. 19. Screenshot of the improved detail view for AI suggestions/responses in our second prototype. The original text and AI suggestion were presented in a layout with two rows, which we then switched to two columns, as shown here.

StudyAlign: A Software System for Conducting Web-Based User Studies with Functional Interactive Prototypes

FLORIAN LEHMANN, University of Bayreuth, Germany
DANIEL BUSCHEK, University of Bayreuth, Germany

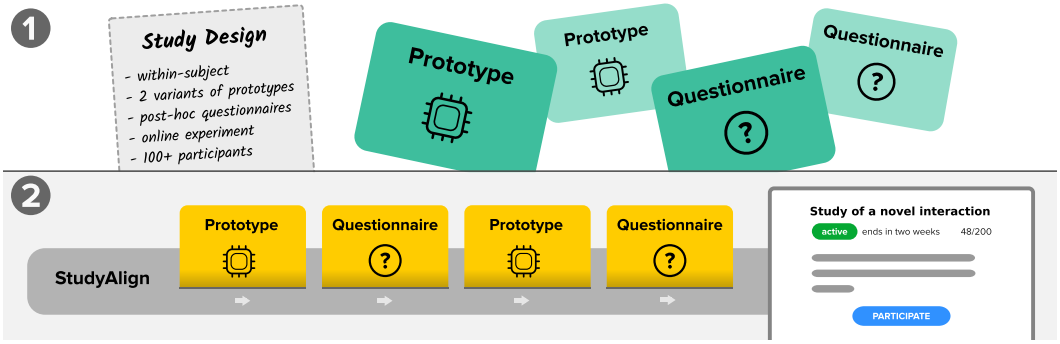


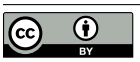
Fig. 1. From 1) a sketched study design to 2) a complete, structured web-based experiment with interactive prototypes: We introduce *StudyAlign*, a software system to conduct web-based experiments with functional interactive prototypes. *StudyAlign* offers four software parts for creating, organizing, and conducting online studies: a Backend to create studies and data logging, a Library to integrate prototypes, a Study Frontend that guides participants, and an administrative Control Panel.

Interactive systems are commonly prototyped as web applications. This approach enables studies with functional prototypes on a large scale. However, setting up these studies can be complex due to implementing experiment procedures, integrating questionnaires, and data logging. To enable such user studies, we developed the software system *StudyAlign* which offers: 1) a frontend for participants, 2) an admin panel to manage studies, 3) the possibility to integrate questionnaires, 4) a JavaScript library to integrate data logging into prototypes, and 5) a backend server for persisting log data, and serving logical functions via an API to the different parts of the system. With our system, researchers can set up web-based experiments and focus on the design and development of interactions and prototypes. Furthermore, our systematic approach facilitates the replication of studies and reduces the required effort to execute web-based user studies. We conclude with reflections on using *StudyAlign* for conducting HCI studies online.

CCS Concepts: • **Human-centered computing** → **HCI design and evaluation methods**; **Interactive systems and tools**.

Additional Key Words and Phrases: software system, software framework, web-based experiments, evaluation, interactive prototypes, interactive AI, user studies

Authors' Contact Information: Florian Lehmann, florian.lehmann@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany; Daniel Buschek, daniel.buschek@uni-bayreuth.de, University of Bayreuth, Bayreuth, Germany.



ACM 2573-0142/2025/6-ARTEICS007
https://doi.org/10.1145/3733053

ACM Reference Format:
Florian Lehmann and Daniel Buschek. 2025. StudyAlign: A Software System for Conducting Web-Based User Studies with Functional Interactive Prototypes. *Proc. ACM Hum.-Comput. Interact.* 9, 4, Article EICS007 (June 2025), 26 pages. https://doi.org/10.1145/3733053

1 Introduction

Imagine you have designed and prototyped novel interaction methods with a functional web prototype. Now, you want to investigate these interaction methods in a user study to collect quantitative and qualitative data. Since your prototypes are built with web technologies, you consider running an online study. However, setting up an online study that combines prototypes (logging quantitative data) with questionnaires (collecting qualitative data), implementing study designs (counter-balancing of conditions and controlling procedures), and conducting the study adds additional effort. Concretely, this effort arises in large parts from building additional software, for example, for data logging backends, implementing the counter-balancing dynamically or through hard coding. Furthermore, these efforts surrounding an experiment tend to re-occur for each study.

In HCI research, user interfaces and interactions are potentially built for the web and examined through online studies (e.g. [24, 29, 41]). A reason to build web-based prototypes is the availability of frontend libraries such as React[1] and Bootstrap[2] that lower barriers for prototyping since many examples and tutorials exist and they are reliable to develop with. Furthermore, researchers have the possibility to run online studies and scale their studies easily. In particular, work on human-AI interaction has recently been based on web prototyping (e.g. [6, 11, 18, 34, 48]). These studies rely on web technologies since they require AI services that are mostly available via the web, for example, platforms like Hugging Face[3] distribute machine learning (ML) models for download, with eased implementation efforts, and instructions for developers. A common approach is to deploy ML models on web servers and make them available via programmable interfaces (APIs). Also, commercial services of such APIs exist. HCI researchers use such APIs to build a model's capability into web application prototypes that enable novel interaction methods or UIs. The process of informing, designing, developing, and analyzing such prototypes typically requires multiple studies to be carried out (e.g. [19, 42, 47]).

Motivated by applied methods from such related work and our own experience in conducting online research, we introduce *StudyAlign*, a software system for conducting web-based experiments. With *StudyAlign*, we follow the philosophy to reduce the previously mentioned efforts surrounding setting up web-based experiments. This way, HCI researchers can focus on prototyping UIs and interaction methods.

Our system is built from four distinct components:

- The Backend: Is the core of the system. It offers database storage for interaction logging and an API to provide important methods to all other components. The Backend ties together conditions, questionnaires, and text pages and offers counterbalancing.
- The Control Panel: Is a browser app that offers a UI for administrative purposes. Researchers use the Control Panel in a web browser to configure and organize their experiments. A crucial feature is the import and export of studies. This feature allows for sharing and replicating experiments.
- The Study Frontend: Is used by participants to take part in a study. The Study Frontend guides participants step-by-step through the experiment and increases control for researchers (e.g. by preventing participants from skipping steps).

[1] https://react.dev/
[2] https://getbootstrap.com/
[3] https://huggingface.co/

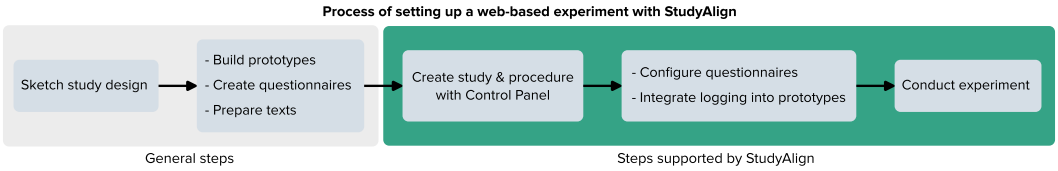


Fig. 2. This figure visualizes the process of researchers for setting up web-based experiments with *StudyAlign*. The first two blocks in this process cover sketching the study design, designing and implementing prototypes, creating questionnaires in tools such as Qualtrics, and preparing text documents, such as task briefings. The latter three blocks are then supported by *StudyAlign*. Researchers use the Control Panel for creating the study and procedure. Instructions support the researchers integrating questionnaires into the procedure, and the Library allows researchers to enable interaction logging in the prototypes. Finally, researchers conduct the experiment by sharing a study link with participants that displays a Study Frontend to guide them through the experiment.

- The Library: Is integrated into prototypes via JavaScript. Researchers include the Library into their prototypes to connect them to the Backend, and log interactions. The Library is an abstraction layer and makes logging easy, for example, logging native browser events.

At the time of writing, the system has been applied in more than ten HCI studies. Some of these studies have already been published (e.g. [6, 10, 11, 15, 55, 56]). Based on this hands-on experience and collaborations with other researchers, we iterated on the features of our system. Furthermore, these deployments demonstrate the system's success in supporting researchers in conducting online experiments.

With *StudyAlign*, we contribute a software system for conducting web-based HCI research. Our system makes setting up such experiments more convenient, allows for interaction logging, and makes sharing and replication of experiments possible directly as files. In this paper, we give detailed insights into how the distinct building blocks of *StudyAlign* work internally and connect to provide the system as a whole. Furthermore, we sketch potential use cases and reflect on previous studies with our system to demonstrate its usability. Finally, we discuss the possibilities offered by *StudyAlign* and the methodological aspects such as replication and study sharing. We make the project available as an open-source project: https://studyalign.com.

2 How to set up an experiment with *StudyAlign*

In this section, we illustrate with a basic example how to set up an experiment with *StudyAlign*. We visualized the process for setting up an experiment as a flow chart in Figure 2. Our example study has a mixed-methods within-subject design. The requirements for this study are as follows: The goal of the study is to compare two different versions of an interaction. Quantitative data should be recorded while participants are using the prototype, and questionnaires should assess the usability and collect feedback from participants after using each prototype. Another questionnaire assesses socio-demographic data and more general feedback. A study design for this kind of experiment is depicted in Figure 9 A, and is a typical design for HCI studies.

Before setting up the experiment with *StudyAlign*, the research team implements the prototypes with their preferred web-technology stack and creates the questionnaires with a survey system of their choice (e.g. Qualtrics). Furthermore, they need to prepare texts for introducing the study to participants, the consent information, and the task briefings.

2.1 Creating a new study

To set up the study, the researchers use the Control Panel, which guides them through four steps. These steps are depicted in Figure 7 and 8. In the first step, the setup wizard asks for general information such as the study's title, dates, the planned number of participants, an introduction text, and a consent text.

In the second step, an interactive view supports the researchers in setting up the study. Concretely, this view allows the researchers to create text elements for the task briefings, condition elements for the prototype versions, and questionnaire elements for the different questionnaires. To add prototypes and questionnaires to the procedures, the researchers have to specify URLs in the elements (e.g. the URL to their hosted prototype web app). Moreover, block elements enable the researchers to group several elements, such as grouping the task briefing, condition, and questionnaire for each prototype version. They can then select these blocks to apply counterbalancing.

The third step gives instructions on integrating the questionnaires with *StudyAlign*. The researchers need to complete these instructions in the survey tools, for example, setting up questionnaire variables that can retrieve data from the URL and save them to the database. For example, in Qualtrics, this is done by following five easy steps.

The researchers can check their setup in the fourth and last step before finalizing the study. Without *StudyAlign* and its Control Panel, the research team would have to technically implement the procedure, supposedly hard-coded by chaining the prototypes, questionnaires, and briefings through chaining links from one study component to another, and take care of realizing the counterbalancing. The more complex a study design gets, the more tedious such manual efforts become, for example, with three prototype variants (i.e. three conditions), or even more complex study designs.

2.2 Integrating interaction logging

After the study has been set up, the researchers need to include the Library into their prototypes and put logging statements at the desired parts in the code, for example, to capture clicks on a certain button. Without our system, the researchers would have to manually create a backend for persisting log data, and implement the requests for calling the backend methods.

2.3 Running the study

The system generates a study link to be shared with the participants. If participants open this link, they see the Study Frontend that guides them through the complete study with a toolbar at the bottom of the screen. Without *StudyAlign*, the researchers would have to think about a way to identify the participants across the procedure, for example, passing an ID through all steps via URLs and implementing when to show the link to the next step, for example, by taking care of a condition's progress. Furthermore, they would have to export the data from their logging backend manually, for example, by writing database queries or connecting to the server to download the logfiles.

To scale a running study, researchers can edit the study and increase the number of participants in the Control Panel. At any time, the study scheme and log data can be exported from the system.

2.4 Summary

Overall, this walk-through shows that *StudyAlign* is connecting many steps in the process of setting up and managing studies (Figure 2), by offering convenient UIs (Figure 7 and Figure 8), and robust methods to research teams.

3 Related work

To inform the design and development of our system, we screened related work on interactive AI prototypes and their evaluation (e.g. [18, 29, 54]). Furthermore, we looked at web-based research and field experiments from a methodological point of view, as well as frameworks and tools for conducting web-based research. In the following paragraphs, we discuss each of these aspects in more detail and how they influenced *StudyAlign*.

3.1 Web-based (field) experiments

The applicability of laboratory and field research in the domain of HCI has been discussed by the research community for decades [51]. Depending on what researchers investigate, one method can be more beneficial than the other [37]. A main difference is the level of control (e.g. control of variables, assignment of participants) and validity these methods offer. Following Shadish et al. [43], validity can at least be further separated into internal validity (causal relationship between manipulated and measured variables) and external validity (generalization of cause-effect relationship). Laboratory studies are commonly considered to offer a high level of control and high internal validity, yet a low external validity. In comparison, field research is considered to offer a decreased level of control and decreased internal validity but an increased external validity. The combination of both methodological approaches results in field experiments. With field experiments, researchers have some control over the experiment while balancing internal and external validity. If participants cannot be assigned randomly, a field experiment is called a quasi-experiment [43]. For the sake of consistency, we stick to "web-based experiment" as a term for "field experiment" throughout this paper.

Such web-based experiments can be conducted online. In other research domains, for example, in psychology, standards for internet-based experimenting were proposed [39], and plenty of software frameworks exist (e.g. [13, 22, 35]). For stimuli-based experiments, comparability of reaction times in lab- and web-based setups was confirmed several times [23]. In a recent survey by Reips [40], web-based psychology research has been reviewed, for example, they discuss web surveys and questionnaire research, web-based tests, and web experiments.

Similar to these experiments from psychology, web-based experiments are a promising approach for HCI research. However, online experiments require the development of technical platforms and tools [36]. In particular, research at the intersection of HCI and AI found the general process of prototyping AI applications challenging [14, 52] but sees a potential for fast prototyping with web-based ML tools [32]. The main challenges in prototyping interactive AI are rooted in the technical complexity [14] and uncertainty [52]. Compared to surveys, interactive AI prototypes are often complex to prototype. Moreover, conducting experiments adds further complexity, for example, maintaining the experiment procedure, realizing a systematic counterbalancing of the order of tasks, and logging interactions.

While our system aims to enable experiments on human-AI interaction, it is not limited to that and supports any web-based prototype. Overall, we introduce a technical basis for conducting web-based experiments with a focus on functional interactive prototypes.

In the following, we examine papers on web prototypes to motivate the requirements of *StudyAlign* and distinguish our system from existing frameworks from psychology, behavioral research, and cognitive sciences.

3.2 Interactive AI prototypes are evaluated on the web

To inform the design and development of our software system, we screened HCI research contributions that investigate interactions with web-based prototypes. In particular, we focused on

research at the intersection of HCI and AI, as these current research efforts rely on web-based prototypes to investigate novel interactions, UIs, and applications. We examined these papers from a methodological point of view. We compared these studies and our own experiences to identify recurring similarities. These similarities defined the requirements and the scope of *StudyAlign*.

Without functional prototypes, yet with web experiments, related work investigated the effects of AI-mediated communication [25] and socially problematic AI-generated email reply suggestions [41]. Web experiments with functional AI prototypes studied text interaction [3, 8, 9, 18, 19, 29, 31, 42, 47], image interaction [12, 34, 46, 50, 54], visualization editing [48], creative applications with sound [33], drawing with language input [24], and the evaluation of a tool to prototype the design of AI applications [45]. Most of these studies consist of multiple parts, such as subsequent studies, for example, to inform and iterate on the design of a prototype, and its evaluation [8, 18, 19, 25, 42, 47, 54]. Data has been collected in all studies, including qualitative data through questionnaires and interviews, and quantitative data through data logging, oftentimes both. For recruiting participants, the researchers relied on platforms such as Amazon Mechanical Turk or Prolific [24, 29, 41]. While related work reported on experiment procedures, the authors were somewhat unclear on how they technically implemented it (e.g. [8, 12, 18, 19, 31, 33, 54]), and randomized condition order instead of applying systematic counterbalancing (e.g. with a Latin Square) [3, 50, 53].

These aspects, namely conducting studies that consist of multiple experiments, interaction logging, online recruitment of participants, and implementation of experiment procedures, defined the core requirements of *StudyAlign*. As implemented, *StudyAlign* enables researchers to set up the procedures of all of these studies (and more). In particular, studies can be configured and managed through an administrative Control Panel, and the experiment logic is provided by the Backend. This enables researchers to run subsequent studies or multiple studies at the same time conveniently (e.g. as in [19, 42, 47]) without re-implementing the experiment logic. Data logging is integrated into the system, as required by all of the mentioned studies, and experiments can be shared on platforms for recruiting participants (e.g. as in [24, 29, 41]). Furthermore, our software system makes web-based experiments more robust by controlling task order with counterbalancing (e.g. as required in [3, 8, 12, 18, 53]). Finally, study designs can be exported and shared easily along with publications.

For the technical implementation, we analyzed similar frameworks and considered their implementation approaches and how we could adapt them for studies with functional interactive prototypes. However, none of these frameworks offered a basis we could extend from, as the next subsection shows.

3.3 Frameworks and tools

Work by Ledo et al. [28] offers a good overview of existing software frameworks and toolkits. Furthermore, the authors identified four common evaluation strategies for such toolkits. Their paper was the initial starting point for our literature research on frameworks, and we also applied two of their identified strategies to evaluate our framework. For our research, we focused on software frameworks that were close to the requirements we derived from papers that evaluated interactive AI prototypes. In particular, we considered experiment frameworks and looked at commercial products and services.

3.3.1 Research frameworks. Most of the software frameworks for conducting web-based experiments stem from research on psychology, behavioral research, and cognitive sciences. We categorized these frameworks primarily by their purpose.

Natural field studies on mobile devices are enabled by frameworks such as *AWARE* [17] and *PhoneStudy* [44]. As a more general tool for experiment design, *TouchStone2* allows researchers to interactively generate and examine trade-offs in experiment designs [16].

Other frameworks focused on backend implementations, experiment application building, and source development kits and libraries: For backend implementations, we found *Pushkin*, a backend aiming for large-scale user studies [20], and *JATOS*, for setting up and managing studies [27]. Another minimalistic backend implementation combined with an experimenter dashboard is offered by *ReActLab*, a framework for programming web-based experiments based on other libraries [5].

The popular Python source development kit *PsychoPy* [38] and a JavaScript library for creating behavioral experiments [13], enable researchers to implement a wide range of experiment application for behavioral sciences, for example, stimuli-based experiments with reaction time tasks.

In general, the larger part of the literature presents frameworks that allow for building interactive experiment applications. These are *Empirica*, a framework to build interactive multiplayer experiments [2], *UILab*, a framework for creating and conducting studies comparing visual designs [7], *SimplePhy*, a builder for perception experiments [26], *OpenSesame*, a graphical experiment builder for social sciences [35], and *lab.js*, a modern online study builder for the behavioral and cognitive sciences [22]. An experiment builder specifically for HCI-related elicitation studies is *Crowdlicit* [1].

3.3.2 Commercial services. In addition to the research frameworks, we also identified commercial online services such as the prototyping tools *Maze*[4] and *ProtoPie*[5]. Both tools offer the setup and management of experiments with prototypes in the design stage. In particular, these tools allow designers to add simple interactivity to their visual designs and combine the prototypes with questionnaires in a procedure. While this approach enables designers to validate design decisions in an early stage of product design, it remains a challenge to prototype interactive AI applications [14], for example, because of the uncertainty and the output complexity of AI's capabilities [52]. In this regard, the research prototyping tool *ProtoAI* was introduced to include AI models in the design stage [45]. The authors of *ProtoAI* conclude, however, that design prototypes may be unable to cover each case that could appear when users interact with AI. They suggest investigating interactive prototypes, sharing them with end-users, and including logging capabilities. In the same line, we argue that functional implementation of these prototypes is required to design, develop, and understand the interaction properly. Furthermore, more extensive studies may be required to understand complex interactions that go beyond design prototypes.

StudyAlign supports researchers with conducting such extensive studies and interactive prototypes beyond the design stage. In the next paragraph, we highlight this concretely and distinguish the scope of our framework from prior work.

3.3.3 The scope of StudyAlign as a framework for web-based HCI experiments. From a technical perspective, *StudyAlign* offers functionalities surrounding web-based HCI experiments. These functionalities are different from already existing systems.

Since functional prototypes are more complex, particularly when they integrate AI capabilities, we do not offer a graphical experiment-building tool, such as [22, 26, 35]. *StudyAlign* offers a library to integrate prototypes into the system and enables interaction logging. It does not offer methods to implement applications specifically, such as *PsychoPy* or *jsPsych* [13, 38]. Creating interactions and building prototypes remains a task of the research teams. However, *StudyAlign* makes it easier

[4]https://maze.co
[5]https://www.protopie.io

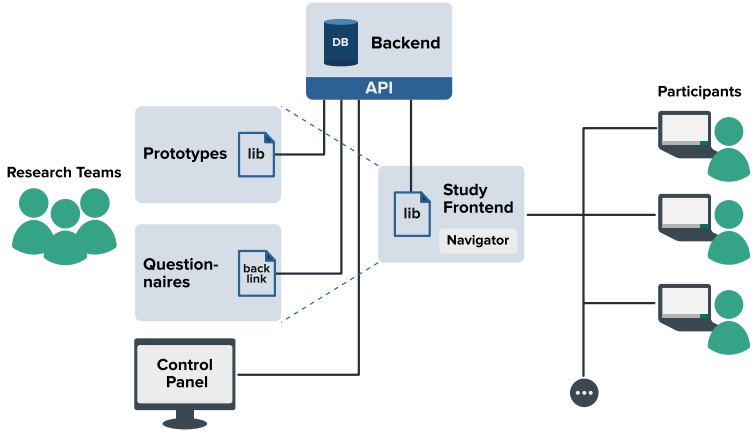


Fig. 3. Overview of the software system *StudyAlign*. The main components are the Backend, Control Panel, Study Frontend, and Library. The Backend is the core of the system since it holds the logic and persists data in a database. The Control Panel is used by researchers to set up and manage studies. The Study Frontend controls the experiment procedure (see Figure 5 about its Navigator for details) and is served to the participants' devices. It embeds prototypes and questionnaires and displays them to participants. Questionnaires use a callback function to work with the API. The Library is the counterpart to the Backend's API and is used in the prototype for interaction logging and in the Study Frontend for controlling the procedure.

for HCI researchers to conduct web-based research with such prototypes, so they can focus on the design, implementation, and analysis of interactions.

4 The *StudyAlign* system

StudyAlign is a completely web-based software system for conducting HCI research. It is not limited to running studies online. It is also possible to utilize it for laboratory setups. In this section, we will state our motivation, give a high-level overview of our system, and provide detailed insights into its software parts. A feature overview can be found in Table 1.

4.1 Motivation

With our software system, we have the overarching goal to advance and substantiate the implementation of methods within web-based HCI research. The initial motivation to start the implementation stems from our experience: we re-implemented experiment logic and logging infrastructure several times across research projects. We found similarities when we compared our experience to related work in our domain. Researchers applied the same methods but gave no insights into implementing the methods. These researchers supposedly also spend time on the development of software and the integration of online services surrounding their research. We identified this as a challenge to provide a foundation for implementing methods in web-based studies. This allows researchers to focus on designing, developing, and studying interactions. Moreover, we support the vision of allowing researchers to share their study design and enable replication.

4.2 Conceptual components

From a conceptual view, *StudyAlign* is composed of software components that closely mirror the conceptual components of experimental methodology. We describe these components and their features and properties in the following.

4.2.1 Studies. A study in our system is the main entity of an experiment and summarizes all corresponding entities (e.g. conditions, questionnaires, log data). A study can be closed access or open access, i.e. private or public. It has a start and end date. Furthermore, researchers can add a description and consent information, such as a privacy policy. For a closed-access study, researchers have to provide a distinct access link to the participants. For an open-access study, everybody who knows the link to the experiment can take part in the study.

4.2.2 Conditions. A condition is an aspect that researchers investigate by conducting an experiment, for example, a prototype that offers a novel UI or interaction method. In other words, each condition is typically one level of an independent variable. A condition in our system can be considered a proxy element that points to a deployed prototype on the web. It is possible to attach a config to a condition. With this config, a prototype can be configured automatically. Researchers only have to deploy one prototype, which reads the current config, for example, to alternate interface elements or interaction methods. In particular, quantitative data is collected with conditions.

4.2.3 Questionnaires. A questionnaire allows researchers to collect subjective data. Similar to a condition, a questionnaire can be considered a proxy element pointing to an external questionnaire service. Identifiers and study IDs are transferred to the survey tool through the questionnaires' GET parameters. After an experiment is finished, it is possible to collect data from the questionnaire service and match it with the *StudyAlign* participant IDs.

Table 1. *StudyAlign* comes with a broad set of features. We highlight the most important features on a system level.

Feature	**Description**
UI for creating web-based studies	The Control Panel supports setting up and managing web-based studies and allows for various study designs. Procedures can be edited with drag and drop, and parts can be selected easily for counter-balancing.
Library for integrating prototypes	The Library can be included in existing prototypes and enables the integration into *StudyAlign* as well as the logging of browser events.
A frontend for participants	The Study Frontend guides the participants through studies from beginning to end and can only proceed one step at a time.
Interaction logging	The system persists log data in the Backend. It supports logging native browser events, such as mouse, keyboard, and touch events. Furthermore, it allows the storage of custom data objects. Methods for data logging are provided in the Library and can be integrated by researchers as "one-liners", to enable easy data logging from inside prototypes.
Sharing of studies	*StudyAlign* allows the import and export of studies. That way, study schemes can be shared easily as material for papers to support future research.
Keeping track of studies	In the long run, research teams can view the history of existing studies and can reuse and evolve their designs.

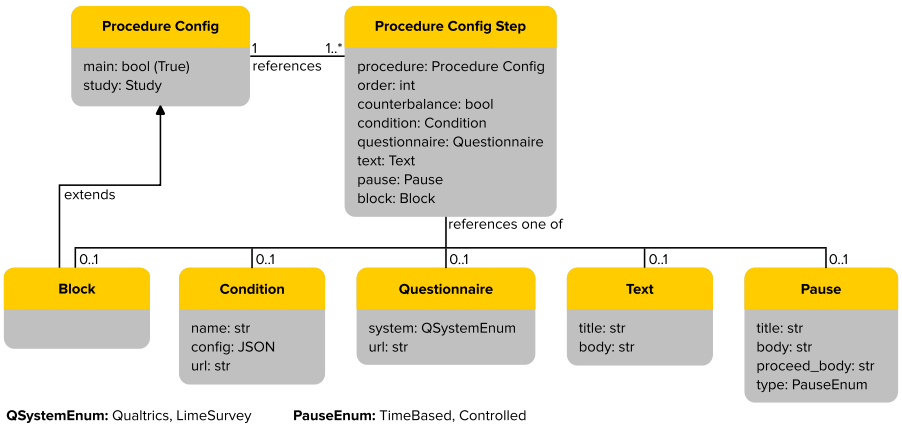


Fig. 4. A class diagram of the Procedure Config used in the Backend to generate procedures. When a researcher sets up a procedure through the Control Panel, a Procedure Config is created. This Procedure Config has multiple Procedure Config Steps. Each Procedure Config Step has an order number and a flag for counterbalancing. Furthermore, it is assigned a concrete procedure element, such as a Condition, Questionnaire, Text, Pause, or Block. A Block extends Procedure Config and can have multiple Procedure Config Steps. This way, a block can combine multiple elements into one element.

4.2.4 Text Pages. A text page communicates information to participants within an experiment, such as a welcome text, task briefing, or general instructions. Text pages allow for HTML markup and embedding of videos.

4.2.5 Pauses. A pause element allows for time-delayed experiments, for example, for interrupted time-series designs [43]. In such experiment designs, an intervention is caused during a procedure to measure its effects. In *StudyAlign*, researchers potentially intervene in the experiment while a pause step is displayed, for example, by modifying a prototype. Researchers can define a concrete information page about the pause, which is displayed to the participants as a text page. A pause can be either time-based or manually controlled. With time-based pauses, the researcher can determine the pause length (e.g. three days). After the predetermined time has passed, participants will be allowed to proceed with the experiment automatically. With manually controlled pauses, researchers have full control over the pause of each participant. Only when an experimenter ends the pause of a participant will be able to proceed.

4.2.6 Blocks. A block is a logical element that allows researchers to combine several other procedure elements into one. In within-subject designs, the procedures might require not only counterbalancing the conditions but also keeping a task briefing and a questionnaire together with each condition. For example, researchers could combine a task briefing, condition, and questionnaire into one block, and then select the block for counterbalancing with other blocks.

4.2.7 Procedures. A procedure represents the generalized experiment procedure. It is an ordered list of subsequent Conditions, Questionnaires, Text Pages, and Pauses. A procedure is defined by the experimenter, who selects which elements are counterbalanced by the system. The system internally uses a procedure config for setting up Procedures, as can be seen in Figure 4. Ultimately, this procedure config is then used to generate a set of procedures, for example, counterbalancing will result in multiple variants of the generalized experiment procedure. These counterbalanced

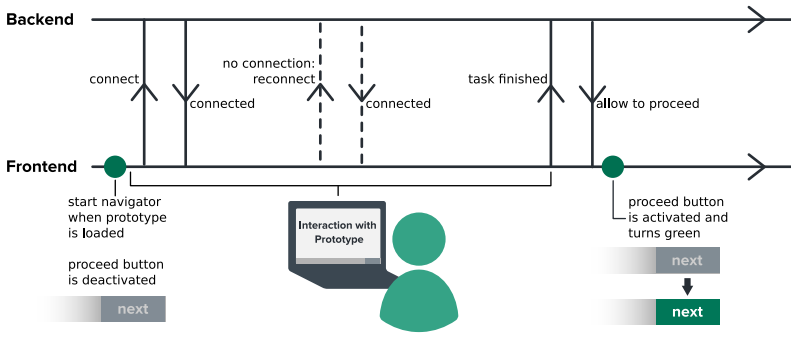


Fig. 5. The Navigator is the conceptual component of *StudyAlign* which controls the procedure. When a participant views a questionnaire or prototype, the Navigator decides whether the participant is allowed to proceed. The diagram is read from left to right and depicts the protocol. The Navigator connects at the beginning of a questionnaire or prototype, tries to reconnect if the connection fails, and waits for a signal when the task is finished for the participant to continue.

variants of procedures are assigned to participants. Further information on typical procedures can be seen in Section 5.

4.2.8 Participants. A participant in our system represents one person taking part in a study. Participants must retrieve an invite link for closed-access studies. A participant has a random UUID, i.e. Universal Unique Identifier, used to uniquely identify a participant and to log that person's interactions. Note that this does not require logging of personal data, the UUID is a random alphanumerical string. As long as researchers do not record identifying data, it is not trivial to reveal the real person taking part in the study. IP addresses are not recorded.

A participant's progress in the procedure is logged automatically. Persisting the progress can be useful, for example, to determine whether a participant dropped out of a study.

4.3 Navigator

On a conceptual level, the Navigator is the part of *StudyAlign* which controls the procedure of a participant. A participant is only allowed to proceed with a study, for example, if the task of a condition is fulfilled. When the participant starts a condition or questionnaire, the Study Frontend connects to the Navigator and listens to the Backend. As soon as the participant is done with the task, a function call from the *StudyAlign* Library from within the prototype updates the Navigator on the server. After that, the Backend sends a signal to the Study Frontend to allow the participant to proceed with the study (e.g. enables a "next" button). The Navigator is implemented as Server-Sent Events for browser compatibility reasons. The flow diagram in Figure 5 shows an example of the Navigator sequence.

4.4 System overview

The architecture of *StudyAlign* is split into four distinct software components: 1) Backend, 2) Control Panel, 3) Study Frontend, and 4) Library. Each component addresses certain requirements of researchers and participants. Figure 3 visualizes how these components interact with each other. Overall, the Backend provides a REST API for communication. All data is stored in the central database of the Backend. The Control Panel and Study Frontend call requests on the Backend's API. The Library needs to be integrated into prototypes, for example, to log interaction data and

make callbacks to the Backend. Questionnaires implement a callback to the Backend to update the procedure state. The research team configures and manages experiments using the Control Panel within a web browser. Participants take part in studies via the Study Frontend on their own devices within a web browser.

StudyAlign components that are implemented as web applications, such as the Control Panel and the Study Frontend, are served by a web server. The prototypes and questionnaires are hosted on the web as well. It is up to the research teams to decide where these components are hosted. It is possible to host them on one machine or split them, for example, for performance reasons. As a practical recommendation, we host the Backend, questionnaires, and Control Panel on one machine. Prototypes run on other instances that allow for more performance, such as servers with access to GPUs to support computationally expensive operations, such as ML model inferences.

We designed *StudyAlign* as a low-opinionated software system. Researchers can choose their preferred tech stack for implementing their web prototype and their preferred web survey tool for implementing questionnaires. We only require that the web prototypes support JavaScript for integrating our Library and that the survey tool passes through GET parameters for including a backlink in the end-of-survey message, to link back to *StudyAlign*.

4.5 Technical components

The *StudyAlign* system consists of four software components. These components are all implemented for web-based applications. However, it is possible to port those components that run on participants' devices to other platforms. The Library and the Study Frontend, for example, could be ported to work in VR applications or on native mobile applications. Next, we describe our implementations of the technical components of *StudyAlign*.

4.5.1 Backend. The Backend offers a REST API for communication and holds the complete experiment logic. It is implemented with Python 3.8. The Backend builds on FastAPI[6], a high-performance web framework for building APIs. FastAPI is compatible with the OpenAPI[7] and JSON Schema[8] standards, generates API documentation with SwaggerUI,[9], and offers data validation with pydantic.[10] The data is stored in a PostgreSQL database.[11] PostgreSQL allows us to combine the benefits of indexed data structures and the flexibility of JSON fields, e.g. for data logging. We use SQLalchemy[12] as an Object Relational Mapper and migrations are carried out with alembic.[13]

Server-Sent Events[14] are used to push data from the server to the client to control the procedure of a participant. Administrative users are authenticated via JSON Web Tokens,[15] while participants are authenticated via Logger API keys in the request headers.

In particular, the Backend processes data, for example, creating and updating data and counterbalancing procedures. Another core function of the Backend is to persist log data. At the moment, the system only supports logging text-based objects. Future updates of the Backend' logging capabilities will also allow storing more complex data, such as images.

[6]https://fastapi.tiangolo.com/
[7]https://github.com/OAI/OpenAPI-Specification
[8]https://json-schema.org/
[9]https://swagger.io/tools/swagger-ui/
[10]https://pydantic-docs.helpmanual.io/
[11]https://www.postgresql.org/
[12]https://www.sqlalchemy.org/
[13]https://alembic.sqlalchemy.org/en/latest/
[14]https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events
[15]https://jwt.io/

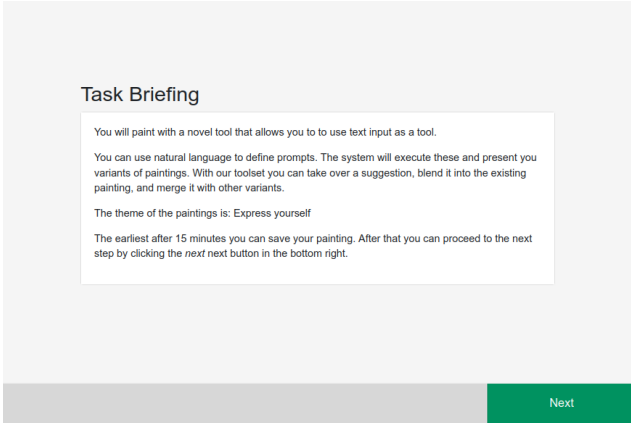


Fig. 6. Screenshot depicting the actual UI of the Study Frontend. In this screenshot, a task briefing that was implemented with a text page element is displayed. The text page displays a title and text instructions. The Navigator UI at the bottom of the screenshot allows participants to go ahead with the procedure by clicking on the "next" button in the bottom right.

4.5.2 Library. The Library drives the Study Frontend and enables interaction logging in prototypes. It is implemented in TypeScript and standardizes the client-side API calls.

For use in prototypes, it provides methods to connect to the Backend and log interactions. It allows logging single interactions or logging batches of interactions to minimize HTTP requests. Native Browser events, such as mouse, keyboard, and touch events, can be logged with minimal effort. For example, this snippet logs a mouse click event:

```
element.addEventListener('click', event => {
  lib.logMouseInteraction('click', event);
});
```

For batched transmissions of interaction logs, events are first logged to the browser's localstorage, and later sent to the Backend in batches of a specified size.

For each event, *StudyAlign* persists the native browser event, the event's timestamp, study id, condition id, participant id, and a custom JSON object. For custom events, *StudyAlign* does not log a native browser event, but any provided JSON object. More examples of interaction logging can be found in the documentation in our project repositories[16].

4.5.3 Study Frontend. The Study Frontend is the user interface that participants see when taking part in a study. It is implemented with React[17] and Redux[18] in ES6. The main function of the Study Frontend is to display the procedure steps and guide participants through a study. The Library is implemented in the Study Frontend for API communication with the Backend.

The UI of the Study Frontend is minimalistic, see Figure 6: The Study Frontend is separated into two areas. The larger area at the top displays text pages, prototypes, and questionnaires. The smaller area at the bottom of the UI displays a navigation bar to allow participants to proceed with the experiment via a "next" button. In conditions and questionnaires, the Navigator maintains a connection to the Backend, as described in Figure 5. The navigator controls the activation of this

[16]https://github.com/StudyAlign
[17]https://reactjs.org/
[18]https://redux.js.org/

button. Activation calls from within the prototypes can request the frontend to activate the next button via the Navigator, for example, when a task has been completed.

4.5.4 Control Panel. The Control Panel is the user interface that researchers and experimenters use to configure and manage studies. It is implemented with React and Redux in ES6. The Control Panel offers a dashboard, a wizard to configure studies and procedures, issuing participant IDs, and the import and export of studies as JSON data, as well as exporting interaction logs as CSV data. An overview of the features offered by the Control Panel can be found in Table 2.

In Figure 7, screenshots depict the first two steps in the UI when configuring an experiment. In these views, experiments are created step by step with a wizard-like interface. Researchers can easily configure each part of a study through forms, for example, by adding general information about the study, as can be seen in the screenshot at the top, and using drag and drop to modify the order of procedure steps, as shown in the bottom screenshot. Figure 8, show the two final steps for setting up an experiment. The screenshot at the top shows instructions for integrating external survey tools, and the bottom screenshot is an overview of input data. Based on the input through this UI, the Backend will receive a formalized version of the procedure, similar to PlanOut [4].

Log data can be exported from *StudyAlign* in the Control Panel. Researchers can then perform an analysis of this data with Python and R.

4.6 External survey tools for creating questionnaires

Our system supports any survey tool to integrate questionnaires into a procedure, as long as the tool supports passing URL parameters to the questionnaire for displaying backlinks. At the moment of writing this paper, we have gained experience with integrating Qualtrics[19] and LimeSurvey[20] questionnaires into experiments. However, other popular survey tools such as SurveyMonkey[21] might support this approach as well: The Study Frontend passes values through the URL to the questionnaire. The survey tool then stores these values to link questionnaire responses with the participants' UUID and puts the values into a backlink in the ''End of survey" message, so that participants are allowed to proceed with the experiment.

[19]https://www.qualtrics.com/
[20]https:/www.limesurvey.org/
[21]https://www.surveymonkey.com/

Table 2. This table provides an overview of the features offered by the Control Panel.

Feature	Description
Overview	The Control Panel offers a dashboard as an overview of all studies.
Manage studies	Studies can be created and modified with a few clicks.
Share studies	Study schemas can be exported with only one click. Researchers can share their studies with colleagues and make them available as material for papers.
Study duplication	Studies can be duplicated. Researchers can adapt from existing studies to run subsequent studies with the same designs.
Log data preview & export	Log data can be viewed from the browser, and also be exported for further analysis in R or Python.

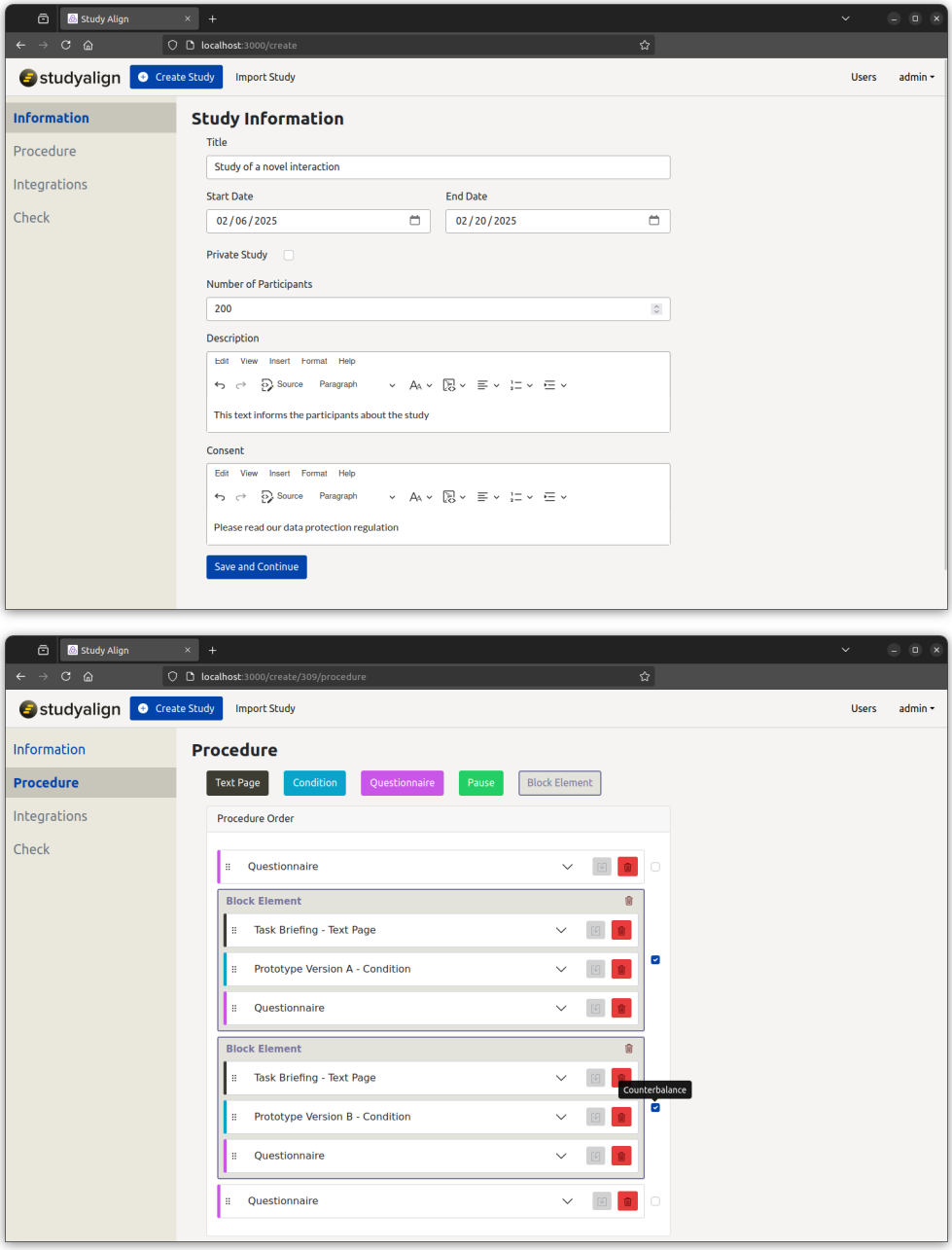


Fig. 7. Two screenshots showing the first two steps of the Control Panel UI for setting up experiments. The screenshot at the top shows the view for researchers to define general information about a study, such as the title, dates, number of participants, descriptive text, and consent text. The screenshot at the bottom shows the view where researchers can interactively compose a procedure. The buttons allow adding new steps which can be sorted through drag and drop. Nesting allows Block Elements to combine several other steps into one Block. Each step can be edited after unfolding the options. Elements that are selected through the checkboxes on the right will be counter-balanced by the system.

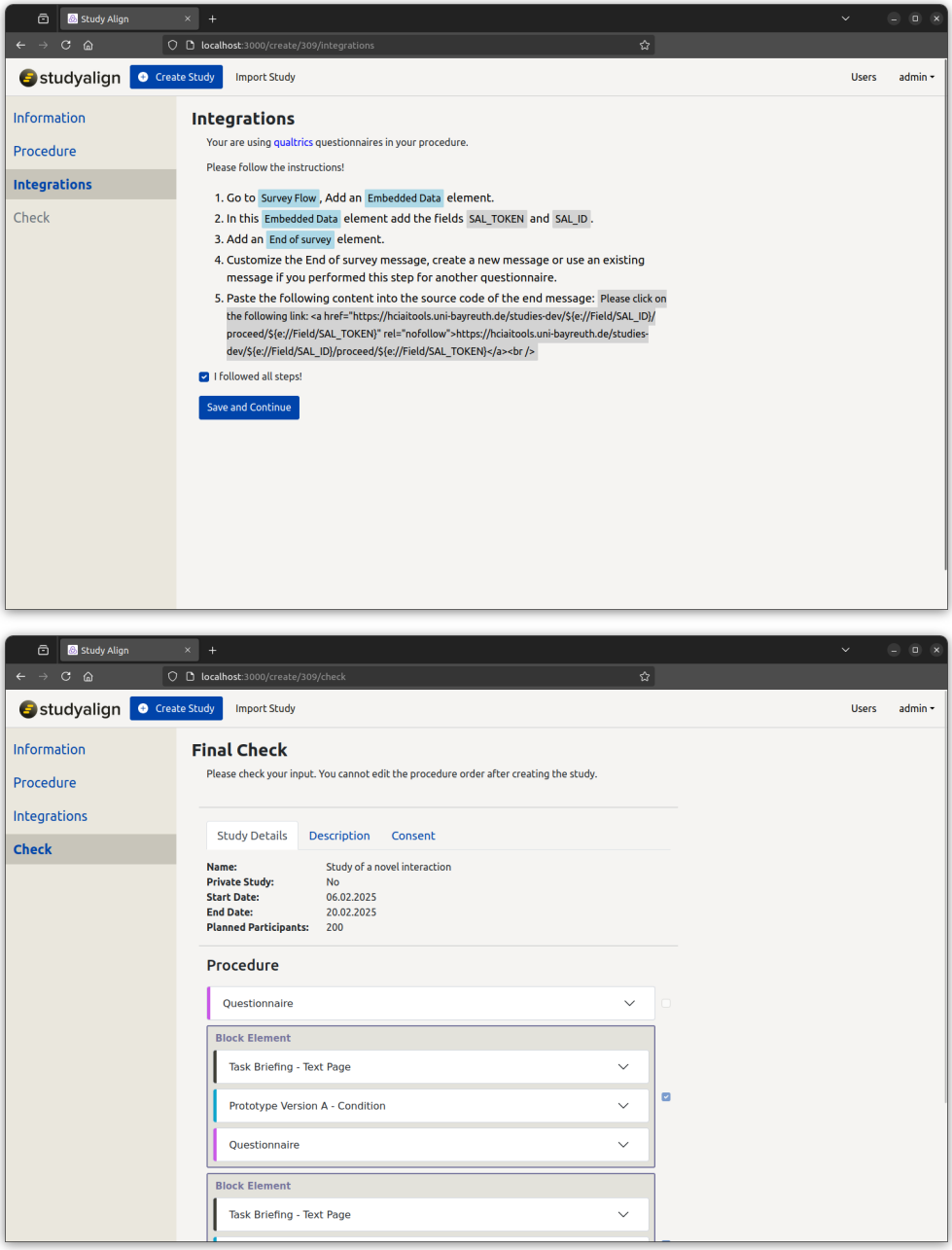


Fig. 8. Two screenshots of the final two steps in the Control Panel UI for setting up experiments. The screenshot at the top displays instructions for researchers on integrating questionnaires into the system. This step is only visible if researchers added a questionnaire to a procedure. The bottom screenshot shows a final overview of all input data so researchers can check if the experiment needs further editing.

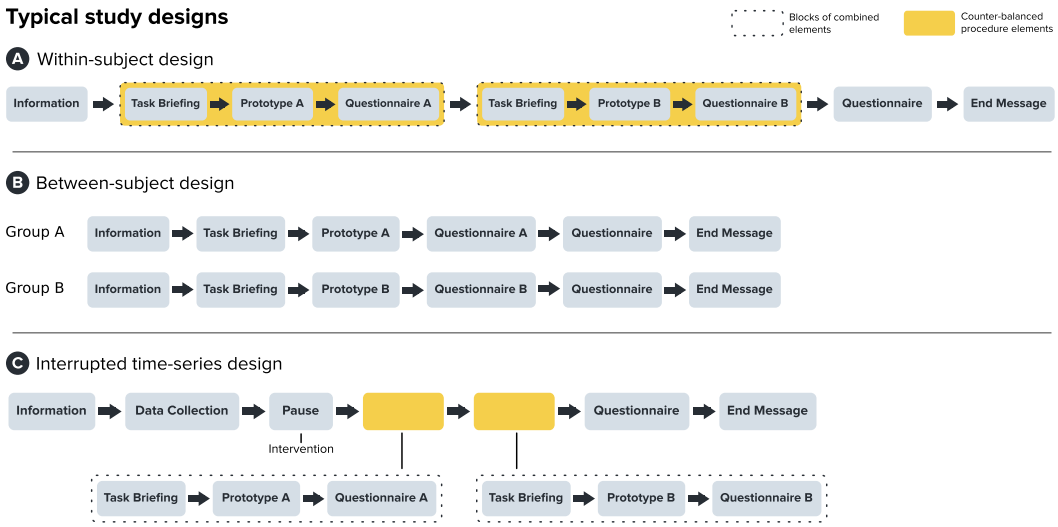


Fig. 9. This figure shows typical study designs that can be implemented with *StudyAlign*. The system supports within-subject, between-subject, and interrupted time-series designs. Example A shows a within-subject design: The task briefings, prototypes, and questionnaires in this example are combined into one "block" element. Such blocks of elements are then counterbalanced. Example B shows a between-subject design: A separate study is created for each group. It is possible to duplicate an existing study, which eases the creation of between-subject designs. Example C shows an interrupted time-series design: The depicted procedure is similar to a within-subject design but with an initial data collection step and an extended pause (e.g. multiple days), for example, to give researchers the time to create an intervention based on the collected data. This intervention then influences the prototypes in the remaining study procedure.

5 Typical Study Designs

In this section, we present commonly used study designs in HCI and how they can be realized with *StudyAlign* on a high level. In Section 6, we present concrete case studies.

5.1 Within-subject designs

In within-subject designs, the same conditions are presented to each participant, but the order of conditions is counterbalanced between participants. For this example, we assume an experiment to compare two user interfaces. At the beginning of the experiment, a briefing is displayed, and prior to each interface variant, a short task description is displayed. After each interface variant, a questionnaire is displayed, e.g. to measure the task load and usability. At the end of the experiment, another questionnaire is displayed to record socio-demographic data and to gather open-ended feedback. See Figure 9 A for a visualization of the concrete procedure. Here, *StudyAlign* counterbalances the order of blocks of prototypes and questionnaires between participants.

5.2 Between-subject designs

In between-subject designs, conditions are compared between groups. Only one condition is presented to people of one group. Typically, there are two or more groups of participants, depending on the number of conditions that should be compared. For this example, we also assume an experiment to compare two user interfaces, but between user groups. The procedure elements stay the same as in the example for within-subject designs, except only one of the user interfaces will

be used and rated by each participant. See Figure 9 B for a visualization of the concrete procedures. Between-subject designs can simply be realized in *StudyAlign* by creating two studies each showing one of the prototypes.

5.3 Studies with an interrupted time-series design

In interrupted time-series designs, the effects of interventions are studied that occur in observations over time [43]. This design can be particularly useful for studies on interaction with AI, for example, when a model is fine-tuned or switched within an experiment.

For this example, we assume an experiment to compare the effects of personalized user interfaces on usability. This example uses a within-subject design with a data collection phase (usage of a baseline prototype) before comparing the two user interfaces. The pause in this example takes two or three days, enough time for the researchers to train an ML model that adapts a user interface according to prior collected user data. At the beginning of the experiment, a briefing is displayed, and before each interface variant, a short task description is displayed. Following each interface variant, a questionnaire assesses the perceived usability. A post-hoc questionnaire collects opinions on the personalized interfaces. After the initial data collection phase, a pause element informs the participant when the experiment continues. See Figure 9 C for a visualization of this procedure. In this example, researchers end the pause for each participant individually and inform the participant via a messaging tool or email to continue. It is possible to configure time-based pauses that count down the time until a participant is allowed to proceed. However, we recommend using time-based pauses only for short windows of time to avoid drop-outs.

6 Case Studies

StudyAlign has been applied in more than ten studies for conducting experiments and collecting interaction data. Six of these studies have been published already [6, 10, 11, 15, 55, 56]. The other works are planned or in the process of being published, for example, [30].

We use these studies as case studies in this section and provide specific details about how *StudyAlign* enabled specific aspects of these studies. All of the presented case studies investigated text-based interactions in web prototypes.

6.1 Simple procedure for a qualitative investigation of a novel interaction method

With *StudyAlign*, an intelligent text editor was evaluated, which helps users to plan, structure, and reflect on the writing process by offering summaries of the text [10]. The authors conducted a qualitative study to gain insights into users' interaction with the prototype. For their evaluation, the researchers implemented a procedure with four steps: 1) an intro to the study, explaining the prototype's features, 2) using the prototype, 3) a semi-structured interview, and 4) a final questionnaire. While taking part in the study, participants had a video call with the experimenter.

All steps were implemented with our system in a procedure as shown in Figure 10 A. The first steps were implemented with text elements that display study information, followed by the task briefing. In the third step, a condition element presented the prototype to the participants and collected interaction data. After completing the task with the prototype, a questionnaire element presented the questionnaire. The procedure concluded with an end message in a text element. By using these elements, *StudyAlign* streamlined this experiment setup and reduced the effort for interaction logging.

6.2 Copy and reuse studies for between-subject designs

This project [30] investigated two alternative versions of a text editor with AI features in two user studies. The initial prototype integrated AI in a conversational UI, while the alternative version

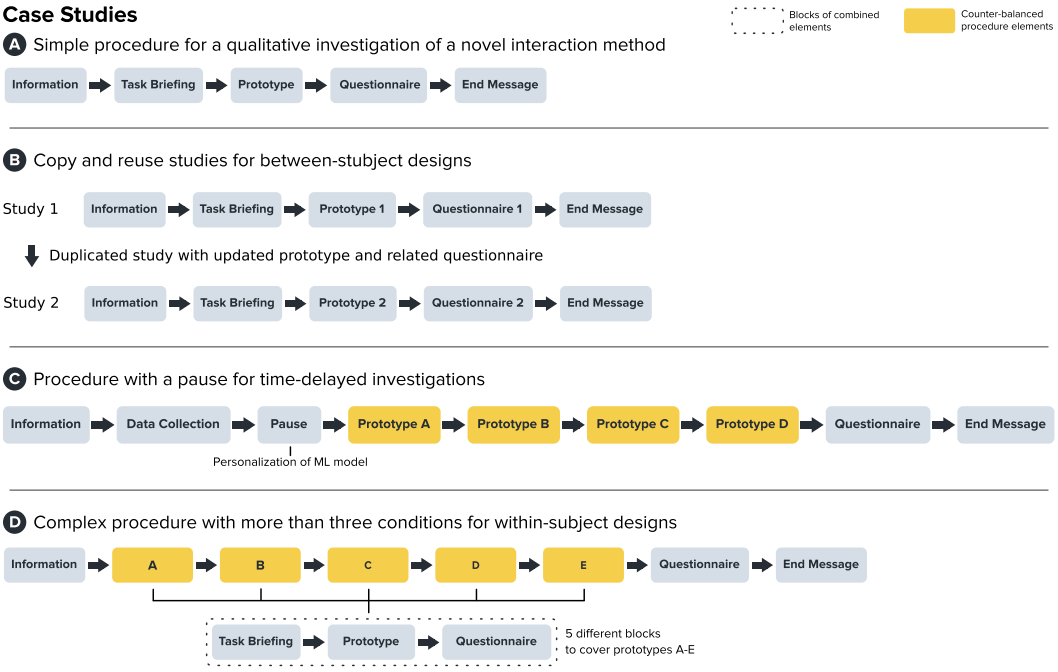


Fig. 10. The procedures of the case studies implemented with *StudyAlign*. These covered qualitative exploratory studies, qualitative and quantitative (mixed-methods) studies, one with experimenter interventions and a time delay (pause), and a comparative study.

integrated AI into a toolbar UI. The researchers implemented the procedure for each study with the following steps: 1) information about the study, 2) the task briefing, 3) prototype usage, and 4) a final questionnaire.

The two procedures of these experiments were implemented with *StudyAlign* as shown in Figure 10 B: they displayed the study information on the initial page, followed by a text element that described the task, including an embedded video. The third step used a condition element to present the prototype and collect interaction data. After completing the task, a questionnaire element displayed a questionnaire. Finally, a text element was used to thank the participant for taking part in the study.

The second study with the iterated prototype was created by duplicating the first study with *StudyAlign*. The only required changes were to replace the URL of the prototype and the questionnaire in the respective elements and to modify the task briefing to explain the new UI.

In summary, the "duplicate" feature of *StudyAlign* made creating an alternative version for the follow-up study easy. The duplicated study could be modified conveniently through the Control Panel.

6.3 Procedure with a pause for time-delayed investigations

Another study [15] investigated the effects of personalized LLMs on authorship. This study required a break as part of its procedure – it was paused for participants while the researchers adjusted the model for text generation. The researchers designed the procedure to start with information about the study, and then began with a questionnaire to collect data about each participant's writing. This

initial data collection was followed by a pause of up to three days. In the meantime, the researchers used the responses on the questionnaire to prepare few-shot prompting for a text model. When this was completed, notifications were sent out to the participants to continue with the experiment. In the remaining study, participants used four different prototypes to write text supported by personalized and placebo-personalized ML models. In the final step of the procedure, participants had to fill out a questionnaire to provide feedback on the prototypes. Prolific was used to recruit participants.

This time-delayed experiment was implemented with *StudyAlign* as shown in Figure 10 C: the initial information was displayed on the start page, followed by a questionnaire element that displayed a questionnaire. After that, a pause element was used to realize an experimenter-controlled pause: as soon as the researchers finished adjusting the generations of the model to match participants' writing styles, they allowed participants to go ahead with the experiment through a click in the Control Panel and notified participants via Prolific. Four condition elements displayed four different AI prototypes. These conditions were counterbalanced using the Control Panel. The final step was a questionnaire, implemented through a questionnaire element, followed by a text element with an end message that contained a backlink to Prolific.

In summary, *StudyAlign* enabled this online study via Prolific while maintaining control of the fairly involved procedure, including participants' progression across multiple days.

6.4 Complex procedure with more than three conditions for within-subject designs

A recent study [11] investigated how prompting influences writing behavior. The experiment started with information about the study, followed by an introduction to the task. In the third step, participants provided socio-demographic data. Afterwards, they used five interaction variants in the prototype, each followed by a questionnaire. A final step asked for general feedback on the interaction methods and thanked the participants.

This experiment was implemented with *StudyAlign* as shown in Figure 10 D: the information about the study and GDPR consent was displayed on the initial page. In the first step of the procedure, a text element provided a written briefing. Then, a questionnaire element implemented the socio-demographic questionnaire. Afterwards, participants interacted with five prototypes, accompanied by a specific task briefing and a specific questionnaire. These groups were built with block elements. The resulting five block elements were counterbalanced by *StudyAlign*. To collect general feedback, a questionnaire element followed at the end, and finally, a text element displayed an end message.

In summary, *StudyAlign* lowered the effort of setting up and conducting this rather complicated within-subject design and its systematic counterbalancing, alongside offering interaction logging.

7 Discussion

We reflect broadly on our software system *StudyAlign* and discuss benefits for research teams, and what we have learned from using it for conducting studies and close collaboration with other research teams. Furthermore, we highlight the opportunities that *StudyAlign* provides to HCI research, considering replicating studies and research transparency.

7.1 Benefits for research teams

We expect that research teams benefit in many ways from using *StudyAlign* as a standardized system for conducting web-based studies. On a high level, we see four aspects:

7.1.1 Keeping track of studies. After running multiple studies with *StudyAlign*, research groups will have a history of studies. This history enables group members to adapt from existing studies

and serves as a knowledge base. The latter is even more important in the long run since group members switch regularly.

7.1.2 Running subsequent studies. Recent web-based studies ran multiple subsequent studies as part of an investigation [8, 18, 19, 25, 42, 47, 54]. *StudyAlign* eases the setup of such subsequent studies through its duplication feature and modification of studies with the Control Panel.

7.1.3 Scaling of studies. StudyAlign allows for scaling studies while maintaining some control, for example, about the procedure. Researchers can modify the number of participants even in a running study. The Study Frontend takes care of guiding participants through studies.

7.1.4 Robustness of methods. As we laid out in our motivation for building *StudyAlign*, we repeatedly re-implemented methods for our HCI studies. With such repeated implementations, it is likely that errors occur, and we had to test these implementations every time before conducting studies. *StudyAlign* has proven its reliability in more than ten studies, providing a reusable and robust system for conducting web-based experiments.

7.2 Lessons learned from conducting web-based experiments with *StudyAlign*

7.2.1 Proven interaction logging from more than ten conducted studies. StudyAlign has been used to run more than ten online studies. The interaction logging worked reliably in these studies, some of which have been published already [6, 10, 11, 15, 55, 56]. Furthermore, consider the case studies in this paper (Section 6). To record interactions, we offer to log single interactions or to write them to the server in batches. The latter has the benefit of reducing HTTP requests and database transactions. We recommend relying on batched interaction logging for experiments that need to log a high number of interactions (e.g. keystrokes). The interaction logging is already implemented in the Library and can be reused across projects. Logging methods are accessed in prototypes through adding simple function calls, see Section 4.5.2 for an example. Thus, the Library effectively reduces programming effort for researchers.

7.2.2 Configuration and management of studies need to be as easy as possible. In the early beginning of developing *StudyAlign*, we only made the API and a brief readme file available and offered no UI for administering studies. In particular, from our first study, we identified the need for a Control Panel and a more detailed documentation. Research teams had to work with abstract methods and execute them manually to reach a specific study configuration. We improved this process with a written step-by-step guide, which informed the UI of the Control Panel to set up and configure a study. With the Control Panel, researchers can now create, configure, and manage studies comfortably from their browsers instead of writing JSON files and executing API requests manually.

7.2.3 Run web-based experiments more efficiently. Our initial motivation to create *StudyAlign* was to enable researchers to set up and manage web-based research easily. By applying our system to concrete research projects, we demonstrated how it can support researchers in this regard and make setting up and running studies more convenient.

Qualitative feedback from our collaborators showed that *StudyAlign* improved efficiency, particularly for people executing studies (e.g. PhD students). One person, for example, gave us the feedback "It is very convenient to have everything in one system. This helped me to carry out the study faster.". In the same line, two other users responded "I found the system and logging backend pretty cool, it made the work a lot easier", and "I think the biggest advantage of the system was that you have a unified user interface for questionnaires and prototypes. You can connect several parts on different websites and it still feels like one study. Also, I found it very good and easy to

assemble the study from the text elements, the questionnaire, and the prototypes, which are then automatically counterbalanced.".

All collaborators asked for features, such as recruitment platform integration and the possibility of adding more participants while the study is running. By using a study link generated by *StudyAlign*, people can be invited, for example, on platforms such as Prolific. We implemented the latter request: Allowing more participants to take part in the study is possible by increasing the number of participants in a study.

Furthermore, they asked for a UI for administrative purposes since they relied on the API for setting up the studies: "At that time there was no GUI, so the handling was a bit difficult at first.". As a result, the Control Panel is now a core feature of *StudyAlign*.

Since we involved external users working with the toolkit for collecting feedback, we cover the toolkit evaluation strategy *usage* as identified by Ledo et al. [28]. We also cover their evaluation strategy *demonstration* since *StudyAlign* has been used to run more than ten studies and thus has proven its feasibility.

7.3 More platforms than web browsers and web-based field experiments: VR, AR, native mobile apps, and lab studies

StudyAlign is based on web technologies, and the types of HCI experiments we built our system for mainly evaluate prototypes implemented as web applications (e.g. [18, 29, 53]). While our case studies (6) showed examples of studies on interaction with text models, it is also possible to work with other media in such prototypes, for example, images, videos, and games. Modern web browsers are powerful and flexible: they support these media types, providing great opportunities for prototyping and experimenting.

However, *StudyAlign* is not limited to web browsers. The clear separation of the Backend and the Study Frontend allows researchers to port our system to other platforms. To port *StudyAlign* to another platform, researchers would only have to port the Library and Study Frontend. To run experiments in virtual reality, for example, the Library could be ported to C# and the Study Frontend to C# in Unity. For AR and native mobile apps, the Library and Study Frontend could be ported to, for example, Android and iOS. In contrast, the Backend and Control Panel are independent of this and thus do not require any changes.

The control and flexibility offered by *StudyAlign* is also available for lab studies. We originally designed the system for web-based field experiments, yet it is possible to use it to run studies in the lab. To conduct a lab study with *StudyAlign*, researchers host the software components on a local network or give access to already hosted instances on the web. The participants then take part in the study via the Study Frontend as usual, but in the lab.

7.4 Transparency by sharing prototype code and study materials as a best practice

StudyAlign facilitates research transparency in two ways: First, the Control Panel enables multiple researchers to collaborate on one experiment. For instance, interdisciplinary teams benefit from this when coordinating joint projects. Second, *StudyAlign* allows for replicating experiments and sharing study designs in a formal way – as structured JSON files. This feature contributes to similar efforts from close fields of research, such as cognitive sciences [2, 20, 21, 27], and enables researchers to share study designs in a formalized way to ease replication.

More broadly, we would like to encourage researchers to share other material, such as prototypes and datasets. Recent work showed that researchers still hesitate to publish their prototypes since they consider them unfinished and see commercial value in their applications [49]. However, we argue that in a field of research that creates prototypes – and analyzes interactions through these – the "default" expectation should be to disclose such methods and artifacts as well. With shared

materials, other researchers will be able to review methods more thoroughly. This change in culture would allow, for instance, building on already existing prototypes and replicating study designs.

With *StudyAlign*, we contribute a first effort in this direction by offering the function to share not only the study design as a file but also to export a complete study, for example, including logged data. Researchers only have to add their prototypes and questionnaires to publish complete material.

7.5 The future of *StudyAlign*

We make *StudyAlign* publicly available as an open-source project and implement it within research projects and collaborations with other researchers. While our personal use motivates us to maintain the project, we expect the project to grow. Typically, extensions add more workload to maintain a codebase. To distribute this workload, we plan to encourage the community to take part in the design decisions and development of the system, for example, by offering tutorials.

On the technical side, we designed the software architecture to be maintainable and modular, and consider it to be well-structured. This makes our system flexible and extensible. For future releases, we plan, for example, to integrate more questionnaire services and recruitment platforms, and to extend the logging capabilities. Furthermore, high-load tests of the logging API are required for large-scale user studies with thousands of parallel participants.

8 Conclusion

We introduce *StudyAlign*, a software system for conducting web-based research, such as online studies and web-based experiments. Our software system consists of several building blocks: 1) a Backend that ties together study designs and offers a central store for interaction logging, 2) a Control Panel for administrative purposes that can be used to configure and organize studies conveniently, 3) a Study Frontend that streamlines procedures and guides participants through experiments step by step, and 4) a JavaScript Library to integrate *StudyAlign* into prototypes, for example, to connect to the Backend API and log interactions.

We derived the architecture of our system from the methodological decisions of related work and our own experience conducting online studies. The system offers control over web-based experiments in general and makes research more efficient since the experiment logic does not need to be re-implemented. The system has demonstrated usefulness in more than ten HCI research projects.

9 Open source

We publish *StudyAlign* as an open-source project to the community. We hope to advance research on novel user interfaces and interaction methods by streamlining web-based experiments in HCI. The project can be found online: https://studyalign.com

Acknowledgments

We thank Jannek Sekowski and Niklas Markert for their outstanding contributions to the development of the control panel. This project is partly funded by the Bavarian State Ministry of Science and the Arts and coordinated by the Bavarian Research Institute for Digital Transformation (bidt). Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) — 525037874.

References

[1] Abdullah X. Ali, Meredith Ringel Morris, and Jacob O. Wobbrock. 2019. Crowdlicit: A System for Conducting Distributed End-User Elicitation and Identification Studies. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing*

Systems. ACM, Glasgow Scotland Uk, 1–12. doi:10.1145/3290605.3300485

[2] Abdullah Almaatouq, Joshua Becker, James P. Houghton, Nicolas Paton, Duncan J. Watts, and Mark E. Whiting. 2021. Empirica: a virtual lab for high-throughput macro-level experiments. *Behavior Research Methods* 53, 5 (Oct. 2021), 2158–2171. doi:10.3758/s13428-020-01535-9

[3] Kenneth C. Arnold, Krysta Chauncey, and Krzysztof Z. Gajos. 2018. Sentiment Bias in Predictive Text Recommendations Results in Biased Writing. In *Proceedings of the 44th Graphics Interface Conference (GI '18)*. Canadian Human-Computer Communications Society, Toronto, Canada, 42–49. doi:10.20380/GI2018.07

[4] Eytan Bakshy, Dean Eckles, and Michael S. Bernstein. 2014. Designing and deploying online field experiments. In *Proceedings of the 23rd international conference on World wide web - WWW '14*. ACM Press, Seoul, Korea, 283–292. doi:10.1145/2566486.2567967

[5] Priscilla Balestrucci, Dennis Wiebusch, and Marc O. Ernst. 2022. ReActLab: A Custom Framework for Sensorimotor Experiments "in-the-wild". *Frontiers in Psychology* 13 (June 2022), 906643. doi:10.3389/fpsyg.2022.906643

[6] Karim Benharrak, Tim Zindulka, Florian Lehmann, Hendrik Heuer, and Daniel Buschek. 2024. Writer-Defined AI Personas for On-Demand Feedback Generation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–18. doi:10.1145/3613904.3642406

[7] Nicolas Burny and Jean Vanderdonckt. 2021. UiLab, a Workbench for Conducting and Reproducing Experiments in GUI Visual Design. *Proceedings of the ACM on Human-Computer Interaction* 5, EICS (May 2021), 1–31. doi:10.1145/3457143

[8] Daniel Buschek, Martin Zürn, and Malin Eiband. 2021. The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–13. doi:10.1145/3411764.3445372

[9] John Joon Young Chung, Wooseok Kim, Kang Min Yoo, Hwaran Lee, Eytan Adar, and Minsuk Chang. 2022. TaleBrush: Sketching Stories with Generative Pretrained Language Models. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–19. doi:10.1145/3491102.3501819

[10] Hai Dang, Karim Benharrak, Florian Lehmann, and Daniel Buschek. 2022. Beyond Text Generation: Supporting Writers with Continuous Automatic Text Summaries. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. ACM, Bend OR USA, 1–13. doi:10.1145/3526113.3545672

[11] Hai Dang, Sven Goller, Florian Lehmann, and Daniel Buschek. 2023. Choice Over Control: How Users Write with Large Language Models using Diegetic and Non-Diegetic Prompting. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–17. doi:10.1145/3544548.3580969

[12] Hai Dang, Lukas Mecke, and Daniel Buschek. 2022. GANSlider: How Users Control Generative Models for Images using Multiple Sliders with and without Feedforward Information. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–15. doi:10.1145/3491102.3502141

[13] Joshua R. de Leeuw. 2015. jsPsych: A JavaScript library for creating behavioral experiments in a Web browser. *Behavior Research Methods* 47, 1 (March 2015), 1–12. doi:10.3758/s13428-014-0458-y

[14] Graham Dove, Kim Halskov, Jodi Forlizzi, and John Zimmerman. 2017. UX Design Innovation: Challenges for Working with Machine Learning as a Design Material. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 278–288. doi:10.1145/3025453.3025739

[15] Fiona Draxler, Anna Werner, Florian Lehmann, Matthias Hoppe, Albrecht Schmidt, Daniel Buschek, and Robin Welsch. 2023. The AI Ghostwriter Effect: When Users Do Not Perceive Ownership of AI-Generated Text But Self-Declare as Authors. *ACM Transactions on Computer-Human Interaction* (Dec. 2023), 3637875. doi:10.1145/3637875

[16] Alexander Eiselmayer, Chat Wacharamanotham, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2019. *Touchstone2*: An Interactive Environment for Exploring Trade-offs in HCI Experiment Design. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow Scotland Uk, 1–11. doi:10.1145/3290605.3300447

[17] Denzil Ferreira, Vassilis Kostakos, and Anind K. Dey. 2015. AWARE: Mobile Context Instrumentation Framework. *Frontiers in ICT* 2 (April 2015). doi:10.3389/fict.2015.00006

[18] Katy Ilonka Gero, Vivian Liu, and Lydia Chilton. 2022. Sparks: Inspiration for Science Writing using Language Models. In *Designing Interactive Systems Conference*. ACM, Virtual Event Australia, 1002–1019. doi:10.1145/3532106.3533533

[19] Han L Han, Miguel A Renom, Wendy E Mackay, and Michel Beaudouin-Lafon. 2020. Textlets: Supporting Constraints and Consistency in Text Documents. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems - CHI '20*. ACM, New York, NY, USA, 13. doi:10.0.4.121/3313831.3376804

[20] Joshua K. Hartshorne, Joshua R. de Leeuw, Noah D. Goodman, Mariela Jennings, and Timothy J. O'Donnell. 2019. A thousand studies for the price of one: Accelerating psychological science with Pushkin. *Behavior Research Methods* 51, 4 (Aug. 2019), 1782–1803. doi:10.3758/s13428-018-1155-z

[21] Felix Henninger, Yury Shevchenko, Ulf Mertens, Pascal J. Kieslich, and Benjamin E. Hilbig. 2022. lab.js: A free, open, online experiment builder. doi:10.5281/ZENODO.597045

[22] Felix Henninger, Yury Shevchenko, Ulf Kai Mertens, Pascal J. Kieslich, and Benjamin E. Hilbig. 2019. *lab.js: A free, open, online study builder*. preprint. PsyArXiv. doi:10.31234/osf.io/fqr49

[23] Benjamin E. Hilbig. 2016. Reaction time effects in lab- versus Web-based research: Experimental evidence. *Behavior Research Methods* 48, 4 (Dec. 2016), 1718–1724. doi:10.3758/s13428-015-0678-9

[24] Forrest Huang, Eldon Schoop, David Ha, and John Canny. 2020. Scones: towards conversational authoring of sketches. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*. ACM, Cagliari Italy, 313–323. doi:10.1145/3377325.3377485

[25] Maurice Jakesch, Megan French, Xiao Ma, Jeffrey T. Hancock, and Mor Naaman. 2019. AI-Mediated Communication: How the Perception that Profile Text was Written by AI Affects Trustworthiness. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. ACM Press, New York, NY, USA, 1–13. doi:10.1145/3290605.3300469

[26] Miguel A. Lago. 2021. SimplePhy: An open-source tool for quick online perception experiments. *Behavior Research Methods* 53, 4 (Aug. 2021), 1669–1676. doi:10.3758/s13428-020-01515-z

[27] Kristian Lange, Simone Kühn, and Elisa Filevich. 2015. "Just Another Tool for Online Studies" (JATOS): An Easy Solution for Setup and Management of Web Servers Supporting Online Studies. *PLOS ONE* 10, 6 (June 2015), e0130834. doi:10.1371/journal.pone.0130834

[28] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–17. doi:10.1145/3173574.3173610

[29] Mina Lee, Percy Liang, and Qian Yang. 2022. CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–19. doi:10.1145/3491102.3502030

[30] Florian Lehmann and Daniel Buschek. 2024. Functional Flexibility in Generative AI Interfaces: Text Editing with LLMs through Conversations, Toolbars, and Prompts. doi:10.48550/ARXIV.2410.10644 Version Number: 1.

[31] Florian Lehmann, Niklas Markert, Hai Dang, and Daniel Buschek. 2022. Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship. In *Mensch und Computer 2022*. ACM, Darmstadt Germany, 192–208. doi:10.1145/3543758.3543947

[32] Na Li, Jason Mayes, and Ping Yu. 2021. ML Tools for the Web: A Way for Rapid Prototyping and HCI Research. In *Artificial Intelligence for Human Computer Interaction: A Modern Approach*, Yang Li and Otmar Hilliges (Eds.). Springer International Publishing, Cham, 315–343. doi:10.1007/978-3-030-82681-9_10 Series Title: Human–Computer Interaction Series.

[33] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J. Cai. 2020. Novice-AI Music Co-Creation via AI-Steering Tools for Deep Generative Models. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–13. doi:10.1145/3313831.3376739

[34] Damien Masson, Sylvain Malacria, Géry Casiez, and Daniel Vogel. 2024. DirectGPT: A Direct Manipulation Interface to Interact with Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–16. doi:10.1145/3613904.3642462

[35] Sebastiaan Mathôt, Daniel Schreij, and Jan Theeuwes. 2012. OpenSesame: An open-source, graphical experiment builder for the social sciences. *Behavior Research Methods* 44, 2 (June 2012), 314–324. doi:10.3758/s13428-011-0168-7

[36] Antti Oulasvirta. 2009. Field Experiments in HCI: Promises and Challenges. In *Future Interaction Design II*, Hannakaisa Isomäki and Pertti Saariluoma (Eds.). Springer London, London, 87–116. doi:10.1007/978-1-84800-385-9_5

[37] Elizabeth Levy Paluck and Robert B. Cialdini. 2014. Field Research Methods. In *Handbook of Research Methods in Social and Personality Psychology* (2 ed.), Harry T. Reis and Charles M. Judd (Eds.). Cambridge University Press, 81–98. doi:10.1017/CBO9780511996481.008

[38] Jonathan W. Peirce. 2007. PsychoPy—Psychophysics software in Python. *Journal of Neuroscience Methods* 162, 1-2 (May 2007), 8–13. doi:10.1016/j.jneumeth.2006.11.017

[39] Ulf-Dietrich Reips. 2002. Standards for Internet-Based Experimenting. *Experimental Psychology* 49, 4 (Oct. 2002), 243–256. doi:10.1026//1618-3169.49.4.243

[40] Ulf-Dietrich Reips. 2021. Web-Based Research in Psychology: A Review. *Zeitschrift für Psychologie* 229, 4 (Dec. 2021), 198–213. doi:10.1027/2151-2604/a000475

[41] Ronald E Robertson, Alexandra Olteanu, Fernando Diaz, Milad Shokouhi, and Peter Bailey. 2021. "I Can't Reply with That": Characterizing Problematic Email Reply Suggestions. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–18. doi:10.1145/3411764.3445557

[42] Oliver Schmitt and Daniel Buschek. 2021. CharacterChat: Supporting the Creation of Fictional Characters through Conversation and Progressive Manifestation with a Chatbot. In *Creativity and Cognition*. ACM, Virtual Event Italy, 1–10. doi:10.1145/3450741.3465253

[43] William R. Shadish, Thomas D. Cook, and Donald T. Campbell. 2002. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Houghton, Mifflin and Company, Boston, MA, US. Pages: xxi, 623.

[44] Clemens Stachl, Quay Au, Ramona Schoedel, Samuel D. Gosling, Gabriella M. Harari, Daniel Buschek, Sarah Theres Völkel, Tobias Schuwerk, Michelle Oldemeier, Theresa Ullmann, Heinrich Hussmann, Bernd Bischl, and Markus Bühner. 2020. Predicting personality from patterns of behavior collected with smartphones. *Proceedings of the National Academy of Sciences* 117, 30 (July 2020), 17680–17687. doi:10.1073/pnas.1920484117

[45] Hariharan Subramonyam, Colleen Seifert, and Eytan Adar. 2021. ProtoAI: Model-Informed Prototyping for AI-Powered Interfaces. In *26th International Conference on Intelligent User Interfaces*. ACM, College Station TX USA, 48–58. doi:10.1145/3397481.3450640

[46] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–26. doi:10.1145/3613904.3642400

[47] Kashyap Todi, Luis A. Leiva, Daniel Buschek, Pin Tian, and Antti Oulasvirta. 2021. Conversations with GUIs. In *Designing Interactive Systems Conference 2021*. ACM, Virtual Event USA, 1447–1457. doi:10.1145/3461778.3462124

[48] Priyan Vaithilingam, Elena L. Glassman, Jeevana Priya Inala, and Chenglong Wang. 2024. DynaVis: Dynamically Synthesized UI Widgets for Visualization Editing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–17. doi:10.1145/3613904.3642639

[49] Chat Wacharamanotham, Lukas Eisenring, Steve Haroz, and Florian Echtler. 2020. Transparency of CHI Research Artifacts: Results of a Self-Reported Survey. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–14. doi:10.1145/3313831.3376448

[50] Thomas Weber, Heinrich Hußmann, Zhiwei Han, Stefan Matthes, and Yuanting Liu. 2020. Draw with me: human-in-the-loop for image restoration. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*. ACM, New York, NY, USA, 243–253. doi:10.1145/3377325.3377509

[51] C. G. Wolf, J. M. Carroll, T. K. Landauer, B. E. John, and J. Whiteside. 1989. The role of laboratory experiments in HCI: help, hindrance, or ho-hum?. In *Proceedings of the SIGCHI conference on Human factors in computing systems Wings for the mind - CHI '89*. ACM Press, Not Known, 265–268. doi:10.1145/67449.67500

[52] Qian Yang, Aaron Steinfeld, Carolyn Rosé, and John Zimmerman. 2020. Re-examining Whether, Why, and How Human-AI Interaction Is Uniquely Difficult to Design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM Press, New York, NY, USA, 1–13. doi:10.1145/3313831.3376301

[53] Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: Story Writing With Large Language Models. In *27th International Conference on Intelligent User Interfaces*. ACM, Helsinki Finland, 841–852. doi:10.1145/3490099.3511105

[54] Enhao Zhang and Nikola Banovic. 2021. Method for Exploring Generative Adversarial Networks (GANs) via Automatically Generated Image Galleries. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–15. doi:10.1145/3411764.3445714

[55] Tim Zindulka, Sven Goller, Florian Lehmann, and Daniel Buschek. 2025. Content-Driven Local Response: Supporting Sentence-Level and Message-Level Mobile Email Replies With and Without AI. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 179, 23 pages. doi:10.1145/3706598.3713890

[56] Tim Zindulka, Jannek Maximilian Sekowski, Florian Lehmann, and Daniel Buschek. 2025. Exploring Mobile Touch Interaction with Large Language Models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 727, 21 pages. doi:10.1145/3706598.3713554

Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship

Florian Lehmann*
Niklas Markert*
florian.lehmann@uni-bayreuth.de
niklas.markert@uni-bayreuth.de
Department of Computer Science,
University of Bayreuth
Bayreuth, Germany

Hai Dang
hai.dang@uni-bayreuth.de
Department of Computer Science,
University of Bayreuth
Bayreuth, Germany

Daniel Buschek
daniel.buschek@uni-bayreuth.de
Department of Computer Science,
University of Bayreuth
Bayreuth, Germany

ABSTRACT

Neural language models have the potential to support human writing. However, questions remain on their integration and influence on writing and output. To address this, we designed and compared two user interfaces for writing with AI on mobile devices, which manipulate levels of initiative and control: 1) Writing with continuously generated text, the AI adds text word-by-word and user steers. 2) Writing with suggestions, the AI suggests phrases and user selects from a list. In a supervised online study (N=18), participants used these prototypes and a baseline without AI. We collected touch interactions, ratings on inspiration and authorship, and interview data. With AI suggestions, people wrote less actively, yet felt they were the author. Continuously generated text reduced this perceived authorship, yet increased editing behavior. In both designs, AI increased text length and was perceived to influence wording. Our findings add new empirical evidence on the impact of UI design decisions on user experience and output with co-creative systems.

CCS CONCEPTS

• **Human-centered computing** → **Text input**; **Empirical studies in HCI**; • **Computing methodologies** → **Natural language processing**.

KEYWORDS

mobile text entry, typing, language model, continuous generations, text suggestions, initiative, control, roles, authorship, deep learning, neural network, dataset

ACM Reference Format:
Florian Lehmann, Niklas Markert, Hai Dang, and Daniel Buschek. 2022. Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship. In *Mensch und Computer 2022 (MuC '22), September 4–7, 2022, Darmstadt, Germany*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3543758.3543947

*Both authors contributed equally to this research.

MuC '22, September 4–7, 2022, Darmstadt, Germany

ACM ISBN 978-1-4503-9690-5/22/09…$15.00
https://doi.org/10.1145/3543758.3543947

1 INTRODUCTION

Text input with touch interfaces, such as mobile devices like smartphones is cumbersome compared to text input on laptop or desktop computers [27, 33]. A basic approach to support the user with touch text input is giving suggestions based on language models [16]. This kind of support ships with the default keyboard software on every smartphone. Third-party keyboards such as *SwiftKey*[1] or *Fleksy*[2] provide even more support. Such keyboards do not only give word suggestions while typing, but have also auto-correction implemented. The main goal of such approaches is to reduce effort of the user by minimizing touch input. The underlying methods and technologies of these features stem from the domain of natural language processing (NLP). One approach for building these models are neural networks. A special form of these neural networks are Transformer networks [34] and can be trained, for example, for question answering, translation, or text generation. From a human-computer perspective, we see a particular potential of models that generate text to influence the way we write.

These generative models could be applied for writing text on mobile devices to further reduce the input effort. However, it remains unclear how to best design user interfaces that empower people to interact with text generation and how such design choices affect the text output as well as the experienced human-AI relationship. This motivates our central research question of this paper: *How do fundamental human-AI roles, as embedded in UI and interaction design, influence text entry behavior, experience, and output?*

To make this measurable, we follow a comparative design and evaluation approach: In particular, we introduce a novel interaction method designed to reverse the "roles" of human and AI writers in the current state of the art (and industry): In this design, the AI continuously adds text word-by-word "live" and puts the user in the role of steering and editing the generation, rather than writing. In a user study, we compare this to both a non-AI baseline with manual text entry and to the common design of AI text as a suggestion list. This comparative experience for users allows us to address

[1]https://www.microsoft.com/en-us/swiftkey *last accessed 10.03.2022*
[2]https://www.app.fleksy.com *last accessed 10.03.2022*

questions regarding perceived authorship, inspiration, efficiency, interaction behavior, and the overall human-AI relationship.

Concretely, we created two prototypes that allow us to compare 1) writing with suggestions of sentence completions and 2) continuously generated text. In both prototypes, the text was generated with the neural language model GPT-2. We conducted a supervised, within-subject online study with 18 participants. They wrote a birthday greeting and a text about a vacation with each of these prototypes. To record a baseline, we also let them write text manually. We measured interactions, collected ratings, and conducted interviews on various aspects of the users interacting with the generative models with our input methods.

Our findings reveal aspects that are important when designing and implementing for interactions with generative text models regarding efficiency, inspiration, and editing behavior. In our comparison, we found that writing with AI, that continuously writes text live, leads to high costs for correcting semantically imprecise generations that did not fit expectations. Despite this higher activity, people perceived themselves as authors less since they edited instead of drafting. In contrast, writing with selecting from a set of AI generated phrase suggestions leads to reduced keyboard input yet higher perceived authorship. In both methods, users perceived an influence of the generated text on the wording. We discuss these results considering the level of control, initiative, and roles.

With this work, we contribute: 1) Two prototypes for writing with AI on mobile devices. In particular, we introduce a novel interaction where the user steers the written text instead of actively writing it. 2) A study comparing these prototypes with an extensive analysis of interaction data as well as subjective ratings and feedback. 3) We give design recommendations based on our findings, for example, that a change in role can be used to make the user believe to be the author of a document, and outline implications for future research.

Alongside with this paper, we make the data and study material publicly available to support future research on human interaction with NLP text models. [3]

2 RELATED WORK

2.1 Writing with Language Models on Mobile Devices

Making writing on mobile devices more efficient and comfortable is a challenging and important topic in HCI [23].

A popular approach to address this challenge is to rely on methods from NLP, for example language models of varying complexities. Simple (letter) frequency models can be used, for example, to build adaptive keyboards to reduce input errors [1, 17]. Recently, language models have also been employed for improving error correction, by automatically predicting where to place an entered correction text [13, 36].

The most widely known application of language models for mobile keyboards are word or phrase suggestions and auto-correction. These techniques are implemented in almost all major soft-keyboards, for example in Apple's standard iOS keyboard[4], Google's *Gboard*[5] or Microsoft's *SwiftKey*.

In particular, the use of *text suggestions* is interesting for our work: Suggestions provide the chance to increase efficiency by enabling users to select a word or phrase instead of typing it. In addition, selecting a suggestion eliminates the possibility of typos. There are several studies on the use of suggestions. Some are more general, not focused on mobile devices, such as a study about the number of suggestions provided during email writing [8]. Others are focused on mobile devices, such as studies on the influence of emotional state on selecting suggestions [15] or suggestion presentation [28]. Other work examined suggesting phrases, in contrast to words [4].

Language models are also used to suggest complete (short) replies: For example, *Smart Reply* [21] analyses an incoming email and suggests three short responses that can be selected and sent without any manual typing.

This wide range of research on language models in (mobile) text entry motivates our investigation here. However, in contrast to the focus on efficiency that is currently predominant in the related work, we focus on effects of interaction design on user experience and output when writing "together" with AI (neural language models). To do this, we make design choices that are not motivated by efficiency, but rather by exploring different human-AI relationships. Ultimately, we envision that these two lines of investigation can be combined, in order to design efficient mixed-initiative [19] text input systems with more explicitly considered user experience and output properties.

2.2 Text Generation with Language Models

Language Models model large text corpora as probability distributions [6]. They are typically trained to predict the next word in a text sequence. Two well-known representatives of such models, that also appear in recent HCI work [8, 25, 32], are GPT-2[6] [29] and its successor GPT-3 [7].

One way to use language models for creating text is by showing their output to the user as suggestions. These suggestions mostly contain single words or longer text phrases. The work of Chang et al. [10] shows that it is also possible to give users "topics" to select instead of directly choosing the full suggestions. This could be especially useful on mobile devices, as there is only limited space to display longer suggestion texts.

A related practical example is Google's *Smart Compose* which displays automatic inline suggestions (one at a time) to complete the current phrase or sentence while writing an email [11].

It is interesting to compare *Smart Compose* with *Smart Reply* (see above) since they embed different roles for user and AI: In *Smart Compose*, the user is the main writer and the AI acts in an assisting role. In contrast, in *Smart Reply*, the AI writes the different short answers for the incoming mail and the user only has to choose (and possibly edit) the preferred one.

[3] Data and study material is publicly available on OSF: https://osf.io/p976a/

[4] https://support.apple.com/guide/iphone/type-with-the-onscreen-keyboard-iph3c50f96e/ios *last accessed 10.03.2022*

[5] https://play.google.com/store/apps/details?id=com.google.android.inputmethod.latin&hl=en&gl=US *last accessed 10.03.2022*

[6] https://huggingface.co/gpt2 *last accessed 10.03.2022*

However, in most of the literature on human-AI writing, the role distribution keeps the user as the main writer and the AI as some sort of helper and/or corrector. Our work in this paper challenges this with an alternative design, in which the AI writes "freely" and the user corrects and edits. Our view here is not that this is a more practical alternative for everyday use. We rather see it as a point of contrast which enables us to explore, quantify and evaluate how human-AI roles can be explicitly manipulated through interaction design and how this affects experience and output.

Nevertheless, our alternative design here is not far-fetched and practically relevant, as recent industry examples show: For instance, a related product is *Flowrite*[7]: Users select a theme and provide keywords for the content of an email, which a language model then turns into a full draft. One difference between this product and our studied design here is that we add the generated text word-by-word (and not at once) to emphasize the impression of the system as "writing". Moreover, this gives users the opportunity to react mid-text (e.g. interrupt generation and edit).

2.3 Impact of Writing with AI

Writing with the help of language models can also impact *what* is written, that is, the output text.

For example, the aforementioned AI text suggestions may not only improve text input, but can also provide incentives for exploring new wording or topics. There are several papers about the use of suggestions in creative writing [9, 31]. The work by Arnold et al. [3] shows that typing with suggestions impact the writing and results in a shorter and more predictable text. Furthermore, the study by Buschek et al. [8] indicates that offering suggestions provokes the use of more "standard" phrases (from a business email context).

Language models do not always have to generate text for direct integration in order to impact the writing. Singh et al. [32] developed a writing tool that analyzes the current content of the text field and offers different kinds of matching stimuli, such as pictures, audio or text. Their user study shows that even without accepting the suggested text, this method helps the users when feeling stuck and inspires their subsequent writing.

Another related study shows that people find it easier to write themselves by observing generated text [30]. In this process, before working on a task, the participants were shown how a system would solve this task with the help of text generation. This made it easier for the participants to solve the tasks independently, as they were shown a possible solution approach through the generation.

Two other examples, where generative models support writing without writing directly, are shown in the work of Arnold et al. [5]. They proposed interaction methods which can be used to change the structure of a sentence in a meaningful way without distorting the basic statement. This approach can be used to get more variation into a longer text. Furthermore, the paper presents a technique that helps the user by formulating questions to be answered in the text in order to define guidelines for writing.

Overall, the related work shows that writing with AI can influence the text (e.g. wording and structure) but also impacts factors that go beyond (e.g. inspiration). These effects beyond text metrics motivate us to investigate another such factor in more detail, namely perceived authorship. Concretely, we compare perception and interaction when writing text with AI in two contrasting designs, that embed different human-AI roles, initiative, and control.

3 CONCEPTS AND DESIGN

We designed the interactions and the UI of our prototypes in an iterative process and then decided on two final, contrasting interaction designs.

3.1 Design Process

The aim of our design process was to identify interaction methods that have the potential to support mobile writing with the use of language models. In contrast to the methods presented above (see Section 2), where the AI is mostly in a helper role, we decided to explore designs with a more active appearance of the AI. For example, the AI might write the text and the user would steer the AI and only edit the outcome when necessary.

We started our design process with brainstorming and sketching in an expert group on possible interaction methods. We also took related work for inspiration into consideration [3, 4, 8, 12].

Following an iterative approach, we conducted multiple rounds of sketching, discussions and collecting feedback among our research team over several weeks.

Two key conceptual directions emerged: First, *Writing with Suggestions* allows the user to choose the text to be integrated from a set of AI generated phrases. This represents the typical suggestion list design direction. Second, in *Continuous Generated Text*, the text is written "live" by the AI, and the user is supposed to only correct the text when it is going into a wrong direction.

The main difference between these two methods is the human-AI relationship in terms of the distribution of "roles": In *Continuous Generated Text*, the AI is the writer and the user acts as an editor. Instead, with *Writing with Suggestions*, the user actively controls the writing process by selecting from the supporting AI's suggestions. For both methods, we decided for an interaction where the user initiates the writing by entering text into a text field. This initial writing should indicate the topic of the text to be the starting text for the AI, but we ultimately removed this in favour of a typical single text entry area (more below). Figure 1 shows sketches of the two concepts.

Our concept and design phase was followed by an implementation and testing phase, as well as a small prestudy which resulted in the final versions of our prototypes. In this prestudy, the participants suggested improvements and were observed by the experimenter.

3.2 Final Designs

Our final design decisions are explained in this section, along with a detailed description of the two implemented concepts. To explain our decisions, we refer to Figure 1 and recall the annotations of this figure to refer to distinct parts of the sketches.

Based on our findings from the prestudy, we decided, for both methods, to remove the separate input field (1a) and combine its function with the main text field (2). Before this change, the user should provide the initial input in the input field (1a). With this

[7] https://www.flowrite.com *last accessed 10.03.2022*

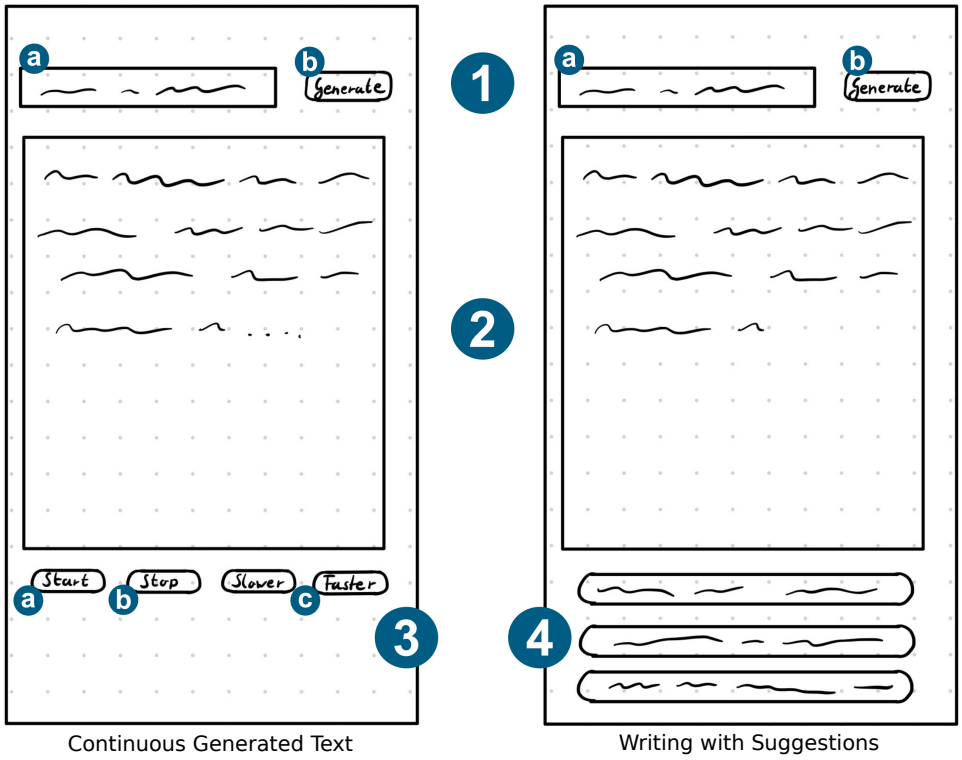


Figure 1: Early sketches of the developed concepts.

change, however, the user simply enters the initial text directly in the main field (2).

3.2.1 Continuous Generated Text. A sketch of the early prototype UI *Continuous Generated Text* is shown in Figure 1 on the left; the corresponding final version is shown in Figure 2 on the left.

The basic idea of this concept was that the user provides the input text and the AI continuously writes a text "live" wordy by word. The user can interrupt the AI while writing and edit the text, for instance, if it does not fit expectations.

The interaction starts with the user entering a sentence in the main text field (2). This starting text can be used to set the context and topic. After that, the AI starts writing: Its generated text appears over time, word by word, in the text field (2). At any time, the user can interrupt the AI by tapping the stop button (3b), for example, when the text deviates from the desired context or the user just wants to make manual improvements to the text. After stopping the AI writing, any parts of the text can be deleted or edited in the text field. When satisfied, the user is free to let the AI continue with writing by tapping on the start button (3a). This procedure can be repeated until the desired outcome is reached.

The initial speed of the appearing text was implemented with one word per second. Thus, without intervention, the AI would write a text with 60 words per minute. That is close to twice as fast as the average typing speed on mobile devices [27]. This should, in the best case, make writing with this interaction method more efficient while still allowing the user to potentially intervene in time.

Based on the feedback from the prestudy and our observations, we removed the buttons to adjust the AI writing speed (3c) and let the AI stop writing when the text field is selected by the user, in order to let the user edit the text.

3.2.2 Writing with Suggestions. Our early prototype for the method *Writing with Suggestions* can be seen in Figure 1 on the right side. The corresponding final version can be seen in Figure 2, also on the right.

The basic idea for this concept was that the user controls the text by selecting AI-generated phrases from a set of suggestions, similar to the use of suggestions in many such systems in related work. By using the language model, these suggestions are based on the text which has been already written.

Similar to *Continuous Generated Text*, the user types an initial text into the main text field (2). The AI generates phrases and presents the suggestions below (4). When the user selects one of these suggestions, it is appended to the text and new suggestions are generated. We added a button below the suggestions to refresh the suggestions manually. This enables users to request new suggestions.

Based on the findings from the prestudy as well as our observations, we allowed the users to manually write and edit text in parallel to the suggestions, instead of writing solely with suggestions. To make the text interaction more natural, we added a delay

of two seconds after the user stopped editing to let the AI present new suggestions.

The number of provided suggestions was informed by related work [8]. This paper shows that with displaying more suggestions, inspiration is promoted, but at the expense of efficiency. We decided to display three suggestions at the same time to balance between efficiency and inspiration. We limited the AI-generated phrases to a length between eight and twelve words. If a phrase completes a sentence, any words behind a terminal punctuation (point, question mark, exclamation mark) are cut off.

4 STUDY

4.1 Design

We used a 3×2 within-subject design to compare three interaction methods in two tasks each. The methods were: *Continuous Generated Text*, *Writing with Suggestions*, and a baseline (manual typing, no AI). We used two writing tasks (see Section 4.4). The combination and order of the writing tasks and the methods was counterbalanced following a Latin-Square [14].

We logged interactions during writing as follows: For each text we recorded the submitted result, the needed time, the number of generations used and for *Writing with Suggestions* also how often a user refreshed the given suggestions. In addition, any interaction with the text, such as typing or deleting a character, was logged along with a timestamp. We also video-recorded most of the sessions with participants' consent.

4.2 Participants

A total of 18 people (14 male, 4 female) participated in the study. All participants were recruited via social networks or email lists. Their age ranged from 18 to 33 years, with a median of 24 years. Originally, we had 20 participants in our study, yet two dropped off due to technical reasons.

All had an English level of A1[8] or higher and were therefore familiar with the English language. One person was left-handed, the rest were right-handed. Regarding writing preferences, 15 participants reported holding the smartphone with both hands and to use both thumbs for writing, the remaining three preferred to type one-handed with the thumb.

On the question if they had previous experiences with generated text, 14 people answered with yes. Most referred to using word suggestions or auto-correction, on their mobile devices (eleven people). A few participants had already gained experiences with generative writing, for example, when composing mails with Gmail or intelligent translators (three people).

Participants were compensated with €10.

4.3 Apparatus

Based on our concepts, two prototypes were realized as mobile web-applications for the study. The frontend was implemented with JavaScript, using the UI library React[9]. For the backend, Python was used together with the web framework Flask[10]. Text generation was realized with the pre-trained language model GPT-2. Figure 2 shows the final UIs.

The prototypes were deployed online. This way, the participants could easily complete all tasks on their own smartphone. Participants took part in the study via a provided link, without having to install software beforehand.

4.4 Task

Participants had the task to write different short texts on mobile devices. We designed two tasks to have a wider range of situations to test the interaction methods: The Birthday-Task is more specific and constraint through exact instructions on what to write. In contrast, the Vacation-Task is more open. We set the minimum length of a text to 20 words, but had no constraints on the maximum length. The texts had to be written in English.

Birthday-Task:	*It is your friend's birthday. Write an email to wish all the best and mention that you have to meet again in some time.*
Vacation-Task:	*Write a short story about your last or upcoming vacation.*

4.5 Procedure

The study was conducted as a supervised online study via video calls. It was separated into three parts. The procedure started with an introduction to the study in line with our institutional procedures. It was followed by introducing the interaction methods and the task briefing. To complete the procedure, we presented a post-hoc questionnaire in the final step. Throughout the process, an experimenter was present to respond to questions or problems. Verbal communication happened mostly in German, with exceptions where the participants preferred English. The complete procedure took 35 to a maximum of 80 minutes, with an average duration of about 50 minutes.

4.5.1 Introduction. At the beginning, the participants were informed about the study and the data protection guidelines, which had to be agreed in order to take part. The two prototypes, *Continuous Generated Text* and *Writing with Suggestions*, were then presented and explained to the participants. In order to get a feeling for the interaction with the prototypes, the participants were allowed to test the two methods voluntarily before starting with the tasks.

4.5.2 Main part. We briefed the participants to complete both tasks, as presented in Section 4.4. For each task, three texts had to be written. One text in each of the two prototypes and one manually with a standard text input for reference. In total, the participants had to write six texts. After completing a text, the participants rated six statements, for example on satisfaction and authorship, on a five-point Likert scale. An overview of the questions is shown in the table in the appendix (Section 8.1).

[8] CERF scale: https://www.coe.int/en/web/common-european-framework-reference-languages/table-1-cefr-3.3-common-reference-levels-global-scale *last accessed 10.03.2022*

[9] https://www.reactjs.org *last accessed 10.03.2022*

[10] https://flask.palletsprojects.com/en/2.0.x *last accessed 10.03.2022*

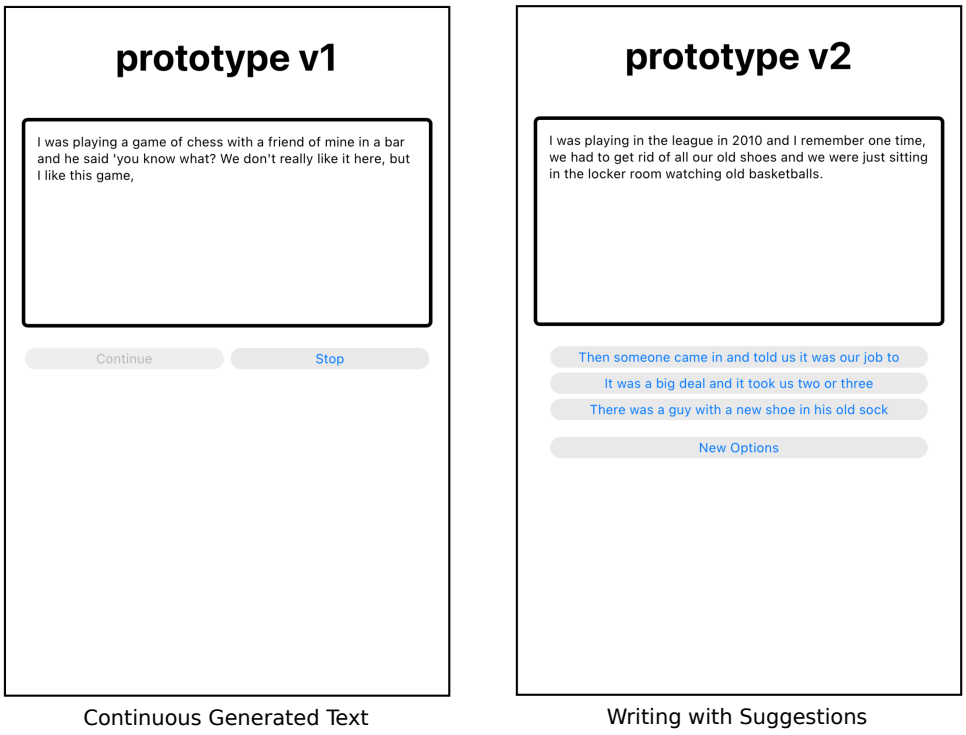


Figure 2: The final versions of the prototypes we implemented as mobile web-applications.

4.5.3 Questionnaire. After completing all six possible combinations of tasks and methods, participants filled in a post-hoc questionnaire. It asked questions on socio-demographics, the interaction methods, and general feedback. An overview of the questions of the post-hoc questionnaire can be seen in the tables in Section 8.2.

5 RESULTS

To analyze our data, we adapted similar approaches from related work [8, 35]. We processed the data with Python[11] and analyzed it with R[12]. We used R mainly for significance testing through linear mixed-effect models (LMMs) and relied for that on the package lmerTest [24]. For the LMMs, the factors *subject* (participant) and *task* were taken into account as random effects, the only fixed effect was the *input method* (interaction method). Based on the LMMs, post-hoc pairwise comparisons were computed with a Tukey test (Z-test) and a Bonferroni–Holm correction applied. For the ordinal Likert data, we applied the Friedman test and the Wilcoxon signed-rank test with the package coin [20]. First, we ran a Friedman test to check for general differences in data. If differences were found, we subsequently ran Wilcoxon signed-rank tests to compare the data pairwise. The Wilcoxon signed-rank test was computed with Bonferroni correction. The significance threshold in our reports is $p < .05$.

[11]Python, https://www.python.org/ *last accessed 10.03.2022*
[12]R: A Language and Environment for Statistical Computing, https://www.r-project.org/ *last accessed 10.03.2022*

Throughout the results, we use the abbreviations *Base* for baseline input (manual standard text input), *Cont* for *Continuous Generated Text* , and *Sugg* for *Writing with Suggestions*.

5.1 Dataset Overview

The produced texts contained overall, 6699 words with a mean of 62.03 (SD 29.55). Each text was written within a mean of 3.04 minutes (SD 1.82 minutes), excluding the time needed for generating text. The system generated text within a mean of 3.14 seconds (SD 2.08). A total of 987 generations were executed in the conditions *Continuous Generated Text* and *Writing with Suggestions*, with a mean of 13,71 (SD 11.90). New suggestions were requested with a mean of 5.14 times (SD 5.96) in the *Writing with Suggestions* condition.

5.2 Words per Minute and Text Length

We computed the word count for the words per minute measure by dividing the character length by five, as suggested by Arif and Stuerzlinger [2]. This result was then divided by the minutes, people needed to absolve the task.

As can be seen in Figure 3a WPM across conditions differed only slightly (M_{Base}: 25.36 WPM, M_{Cont}: 23.02 WPM, M_{Sugg}: 27.61 WPM), differences were non-significant.

Text length was measured in characters. The LMMs had both generative interaction methods as positive predictors, *Cont*: β=78.17,

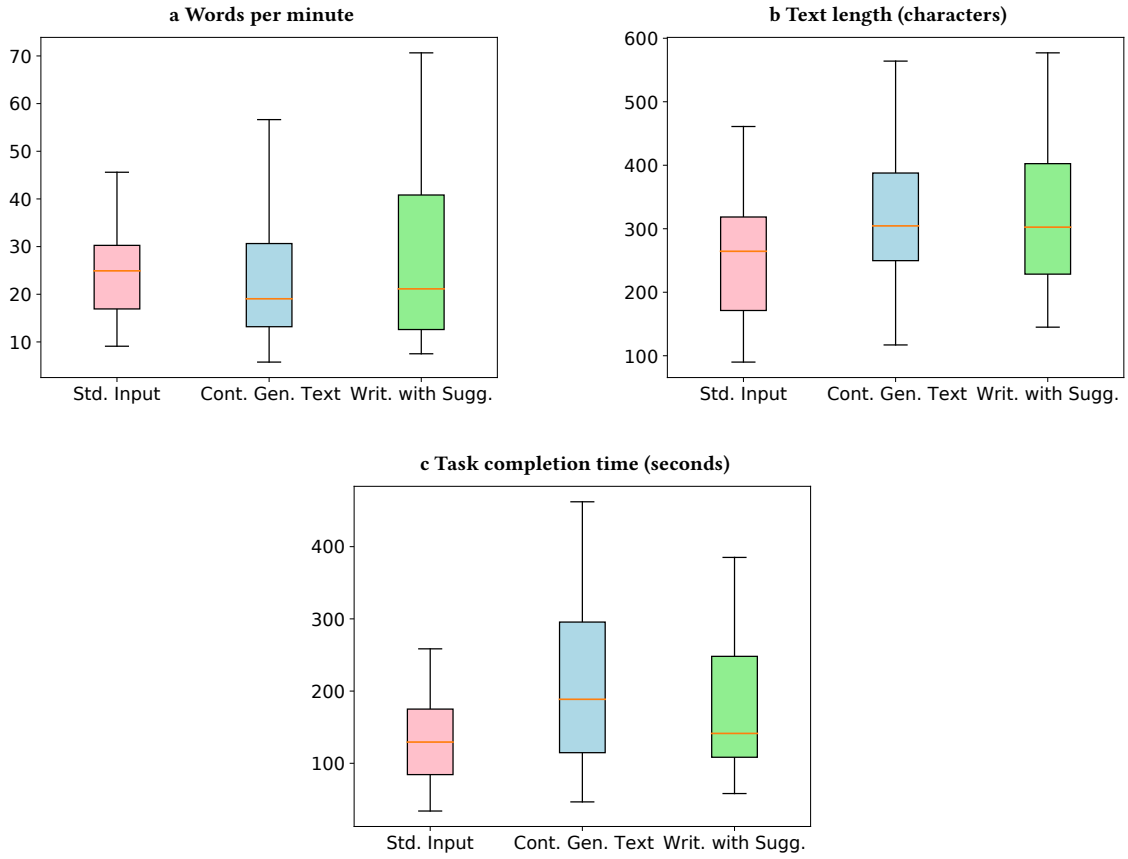


Figure 3: Visualized results of input performance and text metrics a) Words per minute (WPM), b) text length measured in characters, c) task completion time measured in seconds (excluding the time the system needed for generation).

SE=27.095, $CI_{95\%}$=[24.42, 131.91], p = .005 and *Sugg*: β=80.08, SE=27.095, $CI_{95\%}$=[26.34, 133.83], p = .004.

Pairwise comparisons were only statistically significant between the generative interaction methods and *Base*, *Cont* vs. *Base*: Z = 2.89, p = .01, *Sugg* vs. *Base*: Z = 2.96, p = .009. Writing with generative models produced longer text (M_{Base}: 259.64 chars, M_{Cont}: 337.81 chars, M_{Sugg}: 339.72 chars).

5.3 Task completion

We measured the task completion time in seconds from displaying a prototype until completing the task. The time, the system needed to generate the text, was subtracted from the task completion time.

The descriptive overview shows that writing with generative models takes more time than without (M_{Base}: 146.26 s, M_{Cont}: 211.94 s, M_{Sugg}: 189.04 s). Results from LMMs showed the input method to be a significant and positive predictor for task completion time in *Cont*: β=65.68, SE=18.18, $CI_{95\%}$=[29.62, 101.74], p < .001 and *Sugg*: β=42.78, SE=18.18, $CI_{95\%}$=[6.72, 78.85], p = .02.

Pairwise comparisons were only statistically significant between the generative interaction methods and *Base*, *Cont* vs. *Base*: Z = 3.61, p < .001, *Sugg* vs. *Base*: Z = 2.35, p = .049.

5.4 Text Input and Corrections

We measured the keystroke events to analyze the text input and corrections. Since both measures were counts, we fitted generalized LMMs (Poisson family) on them, short GLMMs. We further computed chances of a method influencing text input or corrections for positive β values ($exp(\beta)$) and negative β values ($1 - exp(\beta)$).

5.4.1 Text Input. To analyze text input, we rely on the counts of keyboard strokes per person.

Average keystrokes showed to be minimally higher in *Cont* and lower in *Sugg* compared to *Base* (M_{Base}: 325.08 chars, M_{Cont}: 341.25 chars, M_{Sugg}: 170.91 chars).

The GLMMs had method as a significant predictor. It was a positive predictor for *Cont*: β=.05, SE=.01, $CI_{95\%}$=[.02, .07], p < .001 and a negative predictor for *Sugg*: β=-.64, SE=.02, $CI_{95\%}$=[-.67, -.61], p < .001. The chance of executing more keystrokes is increased by 5.13% for *Cont*. While the chance is decreased by 47.2% for *Sugg*.

All pairwise comparisons were statistically significant, *Base* vs. *Cont*: Z = 3.758, p < .001, *Sugg* vs. *Base*: Z = -40.833, p < .001, *Sugg* vs. *Cont*: Z = -40.833, p < .001.

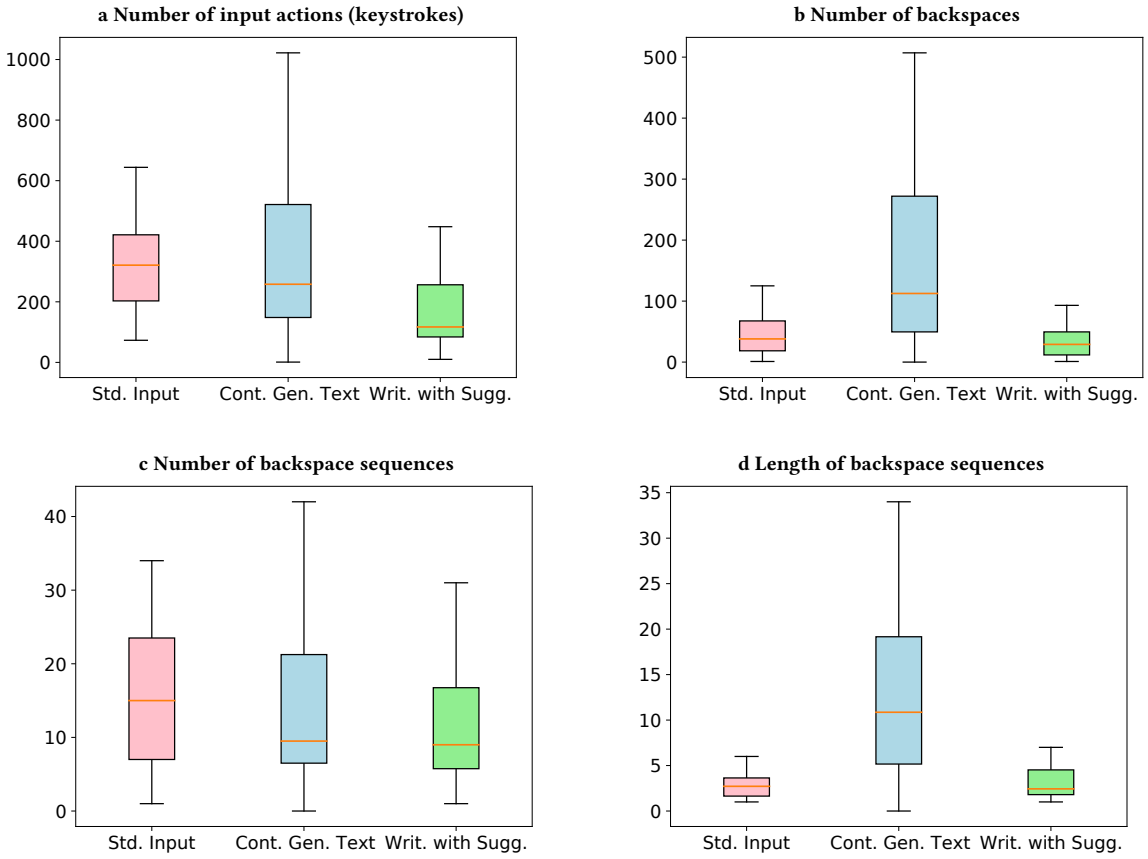


Figure 4: Visualized results of the users' text interactions. a) Number of input actions based on all keystrokes, b) number of backspaces for the purpose of correction, c) number of backspace sequences where a sequence is longer than one backspace, and d) the average length of backspace sequences.

5.4.2 Corrections. To analyze corrections, we took exclusively the backspace keystrokes into consideration. The descriptive statistics show heavily increased corrections (backspaces) for *Cont* and slightly decreased corrections for *Sugg*. (M_{Base}: 45.89, M_{Cont}: 187.83, M_{Sugg}: 39.67). The GLMMs, we fitted on the data, had method as a significant predictor. It was a positive predictor for *Cont*: β=1.41, SE=.03, $CI_{95\%}$=[1.36, 1.46], $p < .001$ and a negative predictor for *Sugg*: β=-.15, SE=.04, $CI_{95\%}$=[-.22, -.07], $p < .001$. *Writing with Suggestions* decreased the chances for text correction by 14%, whereas the chances are almost four times higher (410%) when writing with *Continuous Generated Text*.

The pairwise comparisons were statistically significant for all cases, *Base* vs. *Cont*: Z = 51.376, $p < .001$, *Sugg* vs. *Base*: Z = -4.035, $p < .001$, *Sugg* vs. *Cont*: Z = -53.421, $p < .001$.

In addition, we analyzed backspace sequences. A backspace sequences is defined as multiple backspace keystrokes that happen subsequently. Descriptively, the number of backspace sequences was reduced for generative methods compared to *Base* and the lowest overall for *Sugg* (M_{Base}: 15.86, M_{Cont}: 13.83, M_{Sugg}: 12.39). For the number of backspace sequences, the GLMMs had method as a significant predictor. It was a negative predictor for *Cont*: β=-.14, SE=.06, $CI_{95\%}$=[-.26, -.02], $p = .025$ as well as for *Sugg* : β=-.25, SE=.06, $CI_{95\%}$=[-.37, -.12], $p < .001$. *Continuous Generated Text* decreased the chances for backspace corrections by 13%, with *Writing with Suggestions* chances are decreased by 22%.

The pairwise comparison was nearly statistically significant for *Cont* vs. *Base*: Z = -2.24, $p = .065$ and statistically significant for *Sugg* vs. *Base*: Z = -3.92, $p < .001$.

Moreover, we ran an analysis on the averaged backspace sequence length. Descriptive statistics show a heavily increased sequence length (M_{Base}: 3.04, M_{Cont}: 13.05, M_{Sugg}: 3.60). The LMMs we fitted on the data had method as a significant and positive predictor for *Cont*: β=10.01, SE=1.50, $CI_{95\%}$=[7.04, 12.98], $p < .001$.

In a pairwise comparison, only the comparison to *Cont* were statistically significant, *Cont* vs. *Base*: Z = 6.68, $p < .001$, *Sugg* vs. *Cont*: Z = -6.31, $p < .001$. More text needs to be corrected with *Continuous Generated Text*.

5.5 Perception

People rated their perception on each task via Likert items (see Figure 5). Ratings were given on a scale from one to five, from "strongly disagree" to "strongly agree". For the sake of our analysis,

we concentrated on comparing the methods and therefore average the ratings per person per method. These averaged ratings result in a scale of ten steps instead of five steps. Since only two values were averaged, the mean is equal to the median. The order of the ratings is maintained. To analyze the data, we first ran a Friedman test to check for differences between ratings on methods. When the test result was significant, we ran a Wilcoxon signed-rank test with Bonferroni-Holm correction.

5.5.1 Suitability and Helpfulness. The medians for ratings on suitability were *Base*: 3.5 (IQR = 1), *Cont*: 2.5 (IQR = 1.75), *Sugg*: 3.5 (IQR = 1.375). These differences were statistically significant χ^2 (2, N=18) = 9.5, $p < .01$. Pairwise comparison was only significant for *Cont* vs. *Sugg* (Z = -2.82, $p = .009$). *Writing with Suggestions* was perceived to be more suitable to write text on mobile devices in comparison to *Continuous Generated Text*.

Ratings on helpfulness showed medians as follows, *Base*: 3 (IQR = 0.5), *Cont*: 2.5 (IQR = 1.375), *Sugg*: 3.5 (IQR = 1.375). The test on differences was statistically significant χ^2 (2, N=18) = 7.40, $p < .05$. Again, only the comparison between *Cont* and *Sugg* were significant in pairwise comparisons (Z = -2.74, $p = .013$). Compared to *Continuous Generated Text*, *Writing with Suggestions* was perceived as more helpful when writing text on mobile devices.

5.5.2 Satisfaction and Difficulty. The ratings on satisfaction show medians as follows, *Base*: 4.5 (IQR = 1), *Cont*: 3.0 (IQR = 1.75), *Sugg*: 4.0 (IQR = 1). The test on differences was statistically significant χ^2 (2, N=18) = 16.91, $p < .001$. All pairwise comparisons were significant, *Base* vs. *Cont* : Z = 3.10, $p = .003$, *Base* vs. *Sugg*: Z = 2.63, $p = .027$, *Cont* vs. *Sugg*: Z = -2.41, $p < .043$. Text written without generative models satisfied people the most. Text written with *Continuous Generated Text* was perceived as least satisfying.

Median ratings on difficulty were *Base*: 4.5 (IQR = .5), *Cont*: 3 (IQR = 1.25), *Sugg*: 3.5 (IQR = 1.875). The differences were statistically different χ^2 (2, N=18) = 13.13, $p < .002$. In pairwise comparisons, only the comparison *Base* vs. *Cont* was significant (Z = 3.28, $p < .001$).

5.5.3 Authorship and Wording. For the ratings on authorship the medians were as follows, *Base*: 5 (IQR = .375), *Cont*: 3 (IQR = 1.375), *Sugg*: 4 (IQR = 0.875). The test for these differences was significant χ^2 (2, N=18) = 33.63, $p < .001$. All pairwise comparison showed statistical significance, *Base* vs. *Cont*: Z = 3.73, $p < .001$, *Base* vs. *Sugg*: Z = 3.76, $p < .001$, *Cont* vs. *Sugg*: Z = -3.23, $p < .001$. When using *Writing with Suggestions* most people still perceived themselves as author. They perceived less authorship when writing with *Continuous Generated Text*.

Ratings on how the method influenced the wording of the text showed descriptively following differences, *Base*: 3 (IQR = 1.75), *Cont*: 4 (IQR = 1), *Sugg*: 4 (IQR = 1). These differences were statistically significant χ^2 (2, N=18) = 11.29, $p = .004$. Within the pairwise comparisons, only the comparisons *Base* vs. *Cont* (Z = -2.95, $p = .006$) and *Base* vs. *Sugg* (Z = -2.89, $p = .01$) were statistically significant. Both generative interaction methods were perceived by people to influence the wording of the text.

5.6 Subjective Feedback

In a post-hoc questionnaire, people rated further aspects on Likert scales and provided feedback on open-ended questions. We asked them to rate usability, specific aspects of the generative interaction methods, e.g. generation speed in the condition *Continuous Generated Text*, and inspiration in the condition *Writing with Suggestions*. Regarding inspiration, we asked them two questions, but dismissed one questions since the question contained speculations about the future. All ratings we consider for this analysis are visualized in Figure 6.

5.6.1 Usability and Inspiration. Most of people found both generative interaction methods easy or very easy to learn and use. Easy was rated by five people in each condition (27.78% *Cont* and *Sugg*). Very easy was rated by ten people (55.56%) for *Cont* and by twelve people (66.67%) for *Sugg*.

To assess how much people felt inspired by the generative interaction methods, we asked them to provide ratings on the item "the interaction method inspired me to write things I normally would not have thought of". For *Continuous Generated Text*, nine people (50%) agreed on that. For *Writing with Suggestions*, seven people agreed (38.89%) and three people strongly agreed (16.67%). Both methods can inspire people, however, giving suggestions is more suitable for inspiration.

5.6.2 Ratings of the Methods. We also asked people to rate specific attributes regarding the generative interaction methods. Visualizations can be seen in Figure 7.

For the method *Cont* we asked how people perceived the generation speed. The majority, twelve people (66.67%), found the generation speed just right. Only a small fraction, three people (16.67%) found it a little too slow, and only one person (5.56%) rated for way too slow. In general, the generation speed as sufficient, but could be accelerated a little.

For the method *Sugg* we asked how the number and length of suggestions was perceived. Regarding the number of suggestions, the replies were two-fold. Nine people (50%) found them too less, while the other nine people (50%) found the number of suggestions to be just right. Thus, the number of suggestion should be increased to turn out just right for the majority of users.

The length of suggestions was rated by most people as just right. This rating was given by twelve people (66.67%). Four people (22.22%) found the length a little too long, and only one person found the length way too long.

5.6.3 Feedback on Birthday Task. To write a birthday greeting, 14 people (77.78%) preferred the method *Writing with Suggestions* over *Continuous Generated Text*. People argued to have a better control over the text (13), to be more efficient (3), and found suggestions inspiring (2).

However, there were only three people that would prefer *Writing with Suggestions* also over a standard text input. The 14 people (77.78 %) who preferred the standard input over any method working with generated text, found it easier to integrate personal information (8) and mentioned a better control over the text (7). One person who preferred one of the generational methods reasoned: "I find it hard to come up with new ideas for everyone when greeting them on their birthdays. The generated text provides a source of

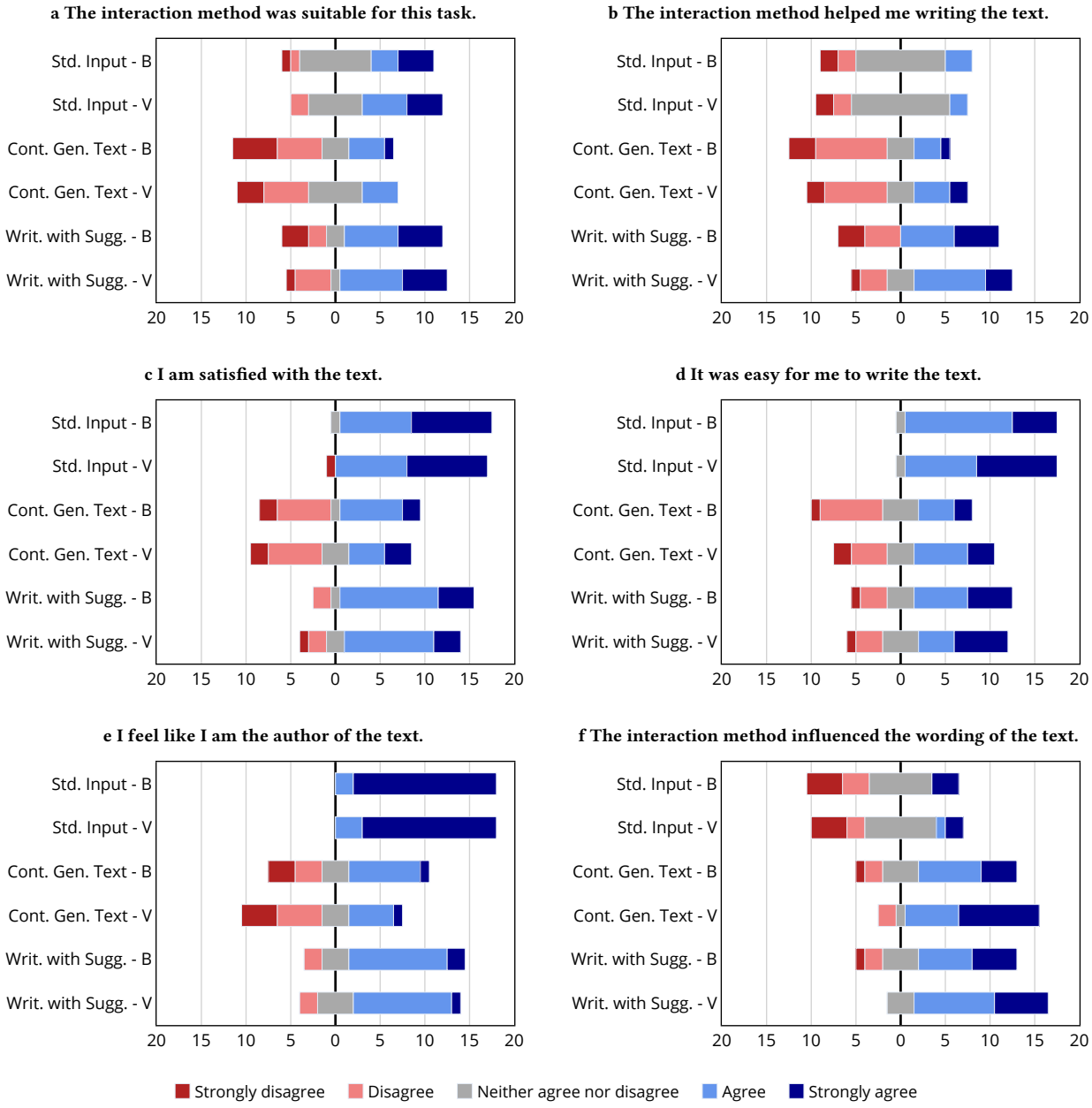


Figure 5: Overview of Likert ratings from the questionnaires we presented to the user after each task. For each method we display two bars: The abbreviation B stands for Birthday-Task, V for Vacation-Task. The x-axes display absolute participant numbers. The legend is displayed below the charts.

inspiration, even when the wording has to be changed by hand.". Besides being inspired from the methods (2) another argument for preferring generational methods was an improved efficiency when writing texts (2).

5.6.4 Feedback on Vacation Task. To write a story about a vacation, twelve people (66.67%) preferred the method *Writing with Suggestions* over *Continuous Generated Text*. One participant, which preferred *Writing with Suggestions*, stated: "The Continuous Generated Text produced more relevant sentences than [in the other task], but still was suggesting things that I didn't really want to tell, in the story. So I prefer Writing with Suggestions, because at least I could somehow control where the story is going." The reasons for preferring *Writing with Suggestions* were therefore mainly the

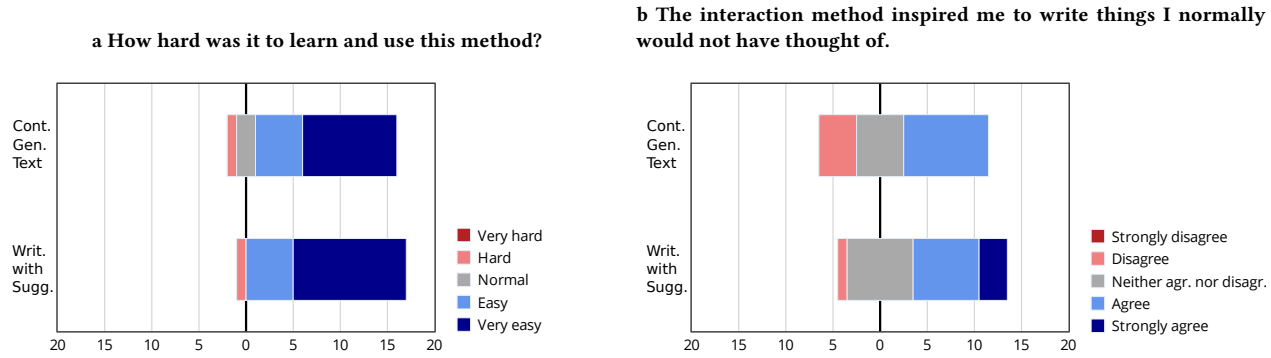


Figure 6: Results of the post-hoc questionnaire on a) usability as a combination of how hard it was to learn and use a method and b) inspiration. X-axes reflect the absolute numbers of participants.

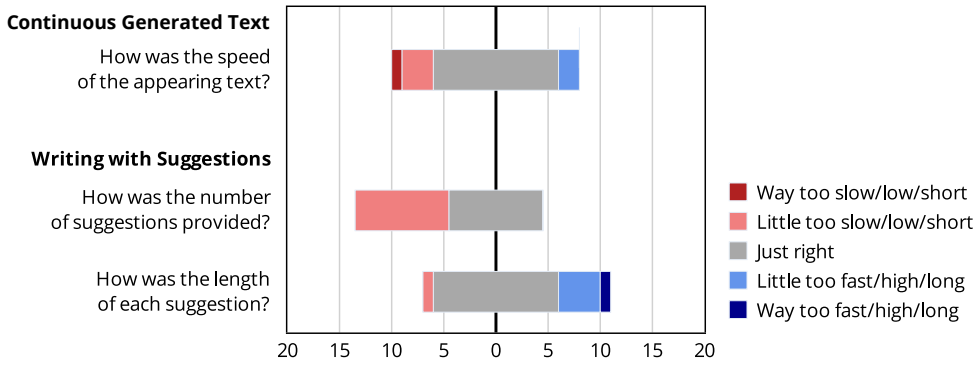


Figure 7: Visualization of method specific results from the post-hoc questionnaire. The top bar shows ratings on the AI writing speed within method *Continuous Generated Text*(legend: "too slow" to "too fast"). The lower bars show ratings on the number of suggestions (legend: "too low" to "too high") and length of the suggestions (legend: "too short" to "too long") within the method *Writing with Suggestions*. The x-axes reflect absolute participant numbers.

better control of the text (7) and the more precise text that could be produced as a result (7).

A total of seven persons (38.89%) stated to prefer one of the generative methods over the standard text field, from which three preferred *Continuous Generated Text* and four *Writing with Suggestions* . The reasons for preferring these methods were mainly the inspiration (3) and efficiency (2). The eleven persons preferring the standard text field argued that, when using the standard text field, the text is more precise and personal (6) and it is easier and faster to write the text (3).

5.6.5 Feedback on the Method Continuous Generated Text. It was suggested that *Continuous Generated Text* could be useful for longer texts (4). In particular, for standard texts, like emails, (7) or some sort of formal texts (7). These are text types that have often a common structure and so give a good opportunity to be automated. Another suggestion that was made are fictional texts (4), where the method has a lot of freedom and also can give a lot of inspiration for writing.

As advantages of *Continuous Generated Text* were mentioned efficiency (7), in order to automate special sorts of text, and inspiration (11), for content and wording of the text.

The main disadvantage of *Continuous Generated Text* is the lack of control over the text (12). This makes it hard for users to include special facts into the text or express exactly what someone wants to say (8). These factors make the method also a bit inefficient to use (4), which is expressed by the answer of one user: "It's slow in the sense that you have to wait quite long before you can see the result. Also long passages have to be deleted by hand, which is also slow and potentially error prone."

According to the users, one feature, which would improve the method, is an optimized language model for creating better text (7). Moreover, they suggested allowing to input special keywords or themes in order to gain more control over the text (7). One participant described these keywords as words that should definitely be included in the text so that they can serve as some sort of guideposts.

5.6.6 Feedback on the Method Writing with Suggestions. The participants mentioned that *Writing with Suggestions* can be useful in a lot of different situations (4), especially when writing longer texts (5). Similar to *Continuous Generated Text*, standard texts (9) and

emails or chat-messages (9) were frequently mentioned as good use cases.

The big advantage of *Writing with Suggestions* were the suggestions. They gave the users the option to always choose the phrases that fitted the most (6), which mostly resulted in useful text (5). Moreover, through the use of suggestions, the users found this method efficient and comfortable (10). Another advantage was frequently mentioned considering suggestions to be inspiring for new topics and different wording (8).

As disadvantages, participants mentioned that the suggestions sometimes were unusable (4), which resulted in long searching times for the right suggestion (5).

For this method, the participants also suggested an optimized language model as a feature. In the best case, this should be user-specific (3) in order to generate better and more suitable suggestions (4). Furthermore, it was mentioned that the suggestions should be shorter (2) and more different (1).

6 DISCUSSION

We discuss differences between the two interaction methods we proposed for writing text on mobile devices. These differences relate mainly to the aspects of control and initiative. Moreover, we discuss efficiency, inspiration and roles. Furthermore, we highlight the perceived influence of the interaction methods on wording and authorship. Finally, we give design recommendations and note research implications.

Overall, we only looked at writing short texts on mobile devices. Findings might deviate for other use cases, for example, other interaction methods or other language models.

6.1 Comparison of the Methods: Control and Initiative Differs, Generated Suggestions Decrease Text Input

With our experiment, we have shown that the design of human-AI interaction influences the level of control and initiative. We have tested two interaction methods in this regard. These methods clearly differ in the way how the generated text is displayed and integrated.

Differences in control and initiative within the presented interaction methods are reflected in the interaction data. Writing with *Continuous Generated Text* increased manual input, while it was heavily decreased for *Writing with Suggestions* (see Figure 4a & Section 5.4.1). With the method *Continuous Generated Text*, the system writes text live word by word. Writing text with *Continuous Generated Text* shifts the focus of the user from writing to editing, and user and system switched initiative in turn. In comparison, with the method *Writing with Suggestions*, we presented three generated completions of sentences to the user. The user was free to decide to take over one of these suggestions. In parallel, text could be manually written by the user. Only the user had the initiative of writing text. The user had full control over the adoption of a particular generated text.

These differences in initiative and control are also visible in the data on text correction. Text written with *Continuous Generated Text* needs correction more often, and backspace sequences are longer than in *Writing with Suggestions* or writing manually (see Figure 4b, Figure 4d & Section 5.4.2). If text gets automatically written by the system, but does not correspond to the user's notion, these generations have to be removed completely up to the point where the user finds the text appropriate. The user has to take back the initiative and exerts control, mostly by editing and correcting AI text. This is prevented in *Writing with Suggestions* since the user can filter the generated text upfront and decides to take over the suggestion. Thus, with *Writing with Suggestions*, the user has the initiative and keeps control through selecting suggestions.

According to the literature on mixed-initiative interaction [19], we contribute two practical examples for mixed-initiative interaction with generative AI. At the same time we challenge the term mixed-initiative since Horvitz defined initiative only implicitly through interfaces that enable efficient collaboration between agent and user. We suggest considering initiative more concretely as a "design material", that is, a variable to explicitly manipulate and reason about when designing such mixed-initiative interfaces. More broadly, we argue that design of intelligent systems should at least consider giving the system the initiative to manipulate content directly to be a truly mixed-initiative design approach. Following this definition, only *Continuous Generated Text* was a mixed-initiative interface, since the system manipulated text directly and was given the initiative by the user. With *Writing with Suggestions*, only the user had the initiative.

6.2 Generative Models can Inspire the User

Our results show that both of our interaction methods helped to inspire the user (see Figure 6b & Section 5.6.1). The results are almost similar, but users agreed on *Writing with Suggestions* to be more inspiring. With *Writing with Suggestions*, we always presented three generated phrases as suggestions. Displaying multiple phrases is more inspirational than displaying automatically written text. This finding is in line with work by [8].

Furthermore, people gave feedback on both interaction methods inspiring them. In order to inspire them, the text did not have to be an exact fit. It was mentioned oftentimes that even if the text did not fit, it gave impulses for new ideas on the topic or an improved wording.

This feedback shows that in order to be inspiring, generated text has not always to be perfect or be taken over as is. The offered text provides the user an idea for a new way of continuing the text. A similar result was found in the work of Singh et al. [32], who called people's (creative) integration of AI contributions "integrative leaps". They showed that various sources of inspiration help the user to keep writing when feeling stuck. In work by Arnold et al. [5], users kept track of the content of the text by giving answers on generated questions. This approach helped writers to make ideas more clear. These two works specifically researched generative AI with the purpose of inspiring the user. In comparison, we found inspiration as a side effect of writing with a generative model. Thus, we argue that inspiration occurs as a general effect when writing with generative AI. In this regard, we see the need for future studies, similar to Singh et al. [32], to explore how inspiring text AI should be integrated into UIs and how they might have to be concretely adapted to certain text types, e.g. formal vs. informal, or writing stages, e.g. drafting vs. iterating.

In general, inspiration is a subjective measure: For example, we asked people if a method was inspiring. It remains a challenge to find objective indicators for inspiration when writing with AI. For this purpose, analytical methods from NLP could be combined with qualitative text analysis to find such indicators. This is another research direction we see for future work on writing with inspiring AI.

6.3 Initiative and Control Should be Integrated as Characteristics in Role Frameworks

We have argued that the differences in initiative and control inherent in each design resulted in measurable changes in interaction. Furthermore, we see an interrelation between initiative and control, and roles when using the methods. We compare the inherent roles of our generative interaction methods to related work on co-creativity. In addition, we look at turn taking within the co-creative process. Before we go into detail for role comparison, we want to clarify that we did not implement an anthropomorphic UI to embody the generative model as an entity. We rather integrated the generative models into the UI as a tool. This clarification is important since some related work on co-creative roles frequently use the term agent – in our case, the generative model (AI) may be exchangeable with the term agent. We rely on three frameworks to classify roles and to reflect on the process of co-creativity in our interaction methods.

First, with the framework by Negrete-Yankelevich and Morales-Zaragoza [26], we attribute the role of a *generator* to both of our methods. Considering this related work, a *generator* is defined by the capability to generate specimens or prototypes of partial or complete pieces of work. Further, we attribute the role of a *master* to *Continuous Generated Text* and the role of an *apprentice* to *Writing with Suggestions*. These roles are underlined by how the methods are intended to work: In *Continuous Generated Text* the system produces a text and the user rather manages and edits the generation, where in *Writing with Suggestions* the system produces a set of suggestions and the user has to pick the best.

Second, the framework by Kantosalo and Toivonen [22] allows for a more fine-grained distinction between roles and capabilities. Regarding this framework, we would attribute the role of *incomplete agents* to both methods, since they are capable of identifying a task and generating text, yet they do not have the capability of evaluating output. Further, we would attribute *Continuous Generated Text* as taking part in *alternating co-creativity*. The authors of the framework argue that an *incomplete agent* cannot take part in alternative co-creativity. However, based on our experiences in this work, we think the missing criteria of a system to evaluate its own output will only cause a drop in the quality of the system's output, but the ability to participate in alternating co-creativity is preserved. For *Writing with Suggestions* we attribute the role of *task-divided co-creation*, since human and system do not take equal turns, instead the system only contributes suggestions. Finally, we use the framework to distinguish between the way an agent contributes to co-creativity. We consider *Writing with Suggestions* as *pleasing* since it conforms more to the user's intent, and *Continuous Generated Text* as *provoking* since it is challenging the user. The latter is also reflected in the ratings on helpfulness as seen in Figure 5b). *Continuous Generated Text* was perceived as less helpful (more challenging) and *Writing with Suggestions* as more helpful (more pleasing).

Third, with the framework by Guzdial and Riedl [18], we depict the process of co-creativity between human and the generative model: In the case of our prototypes, the users are people who are able to write text on smartphones. The AI is a generative model but static, it does not learn by taking into consideration previous interactions. The first turn of the interaction is taken by the user by starting writing the text. With *Continuous Generated Text* the turn is then given to the AI by the user when the system generation is started. The system executes an *artifact action* by writing text word by word, the user at the same time observes passively the system's output. The user can take the turn again if text deviates too much from expectation by stopping the generation. In an *artifact action* the user can edit the text, the AI does not take any actions in the meanwhile. The user's turn ends again by starting the continuously generative writing again. In contrast, in *Writing with Suggestions*, the user always takes turn and executes *artifact actions* by writing text. The AI executes only passive actions by suggestion completions of sentences. Selecting and taking over one of these suggestions is *another artifact action* by the user. With both methods, the turns finally end if the user's expectations on the text outcome are met.

In all of these frameworks we find the dimensions of control and initiative implicitly again, be it in roles, e.g. master vs. apprentice, or interaction, e.g. alternating co-creativity, task-divided co-creation, or turn taking. Based on our experiences and this reflection, we propose to adopt *initiative* and *control* as explicit dimensions in any framework designed to determine roles. Overall, we see the need for a role and process framework for co-creative interaction with generative AI to inform future research and design directions.

6.4 Perceived Quality and Wording Change with Text Written by Generative AI, Authorship is Preserved with Suggestions

Using the generative interaction methods, people were less satisfied with the outcome of the text (see Figure 5c & Section 5.5.2). We interpret this rating on satisfaction as a drop in text quality. Furthermore, texts got longer when writing with AI (see Figure 3b & Section 5.2). We suspect this extended length grows proportional, the more text is written with generative AI. Furthermore, we suspect the extended length causes problems for formal text with a focus on information, since this additional text might cause a drop in information density overall. Our findings in this regard deviate from those by Arnold et al. [3]. They reported that writing with suggestions makes text shorter. This might be true when writing image captions (their case), i.e. very short texts, with suggestions that are only one word long. However, suggestions can vary in length, for example, we employed suggestions with a length from eight to twelve words. We assume that integrating those partial sentences introduced more words than users would have intended in manual writing, and thus made text longer. This finding shows that we cannot generalize the statement by Arnold et al. that writing with suggestions makes text longer. To refine the statement: The final text length depends on the length of adopted AI-generated

text (e.g. suggestion length) and the text format. As a result of reduced satisfaction on text quality and the extended text length, we recommend to revise a text written with generative AI at least once.

Overall, the texts did not only get longer, people also perceived a change in wording when writing with AI-generated text (see Figure 5f & Section 5.5.3). However, when writing with *Writing with Suggestions*, people still perceived themselves as authors. This perception of authorship can be considered a contradiction, since people also had the least text input interactions with *Writing with Suggestions*. People realize that taking over generated suggestions changes wording, but do not realize it might also change "authorship" in terms of the length of human vs AI contributed text. We suspect that a high level of control over accepting generated text and having the initiative in writing are the reasons for this perception. We suggest future studies to investigate more fine-grained levels of authorship, for example, asking if users would attribute co-authorship to the system.

6.5 Design and Research Implications

Based on insights from our analysis on perceived authorship, we suggest raising awareness for authorship in generated text for users or use-cases where this is valued. Text that was generated by AI could be highlighted, for example, by changing font color or adding a background color behind the phrases. Visualizing the ratio of own and generated text would be another option in this regard. This information could be displayed next to other statistics, for example, next to the word count.

Furthermore, we recommend to make suggestions configurable, based on the ratings on suggestion length (see Figure 7). The users could then adjust the number of displayed suggestions, and also the length. In addition to that, suggestions should adapt to context. This context could be, for example, the writing stage. In an idea finding phase, the suggestions would be shorter and more diverse. After the idea finding phase, the suggestions could be longer and more informative. These adaptive suggestions would only be constrained by configurations. Moreover, such adaptions could be extended to the interaction design. The interaction methods could switch, for example, in an ideation phase *Writing with Suggestions* could help with finding ideas and when ideas are found, *Continuous Generated Text* could help to extend text. Another possibility could be to adapt the input methods to the roles, for example, when the user is actively writing, suggestions could be displayed, when the user gets more passive, the system could continue with writing. How to apply these configurations and adaptions, however, requires more research.

Considering our findings on text correction, we recommend to design control options for interacting with systems that automatically write text, e.g. to make the generation more directly steerable. For instance, a function to rewind the written text could be an option for the method *Continuous Generated Text* to avoid long backspace sequences (see Figure 4d).

Our prototypes generated text with GPT-2. Similar to Singh et al. we assume an improved quality of generated text by future models may influence how users interact with and perceive generative systems. This motivates further studies with other models.

In general, our interaction designs demonstrate diverging pathways of how text can be written with generative AI. Assuming further advances in the performance of language models, we expect that, on the one hand, generative text models will replace tedious routine writing (e.g. emails sent to agree on a meeting). On the other hand, we see the potential to aid human writing skills for more (co-)creative tasks (e.g. AI-supported novel writing). In a broad outlook, working on tasks together with generative AI will likely influence the future working life and productivity.

7 CONCLUSION

We have examined writing with AI text generations on mobile devices: We compared two distinct methods in which users and AI can both contribute text when writing on mobile devices. Concretely, we built two prototypes that demonstrate 1) writing with AI generated suggestions and 2) a continuously writing AI. We provide the following findings as answers to our research question on how fundamental human-AI roles, as embedded in UI and interaction design, influence text entry behavior, experience, and output.

Based on our findings, we propose to take initiative as an explicit design dimension into consideration when designing and implementing human-AI interaction to result in truly mixed-initiative applications. Furthermore, we identified control and initiative as possible characteristics for frameworks that conceptualize human-AI roles and propose to extend such frameworks by these dimensions. Our analysis of text entry and behavior showed that people wrote less actively with AI suggestions, but still perceived themselves as authors. Instead, continuously written AI text reduced this feeling of authorship, while effort for editing increased. Overall, the AI generations increased text length and were perceived to influence wording. These results together yield our key conclusion and contribution to the literature: Users' perceived authorship of output, when co-created with AI, is not a simple function of the number of user actions. Instead, the types of these actions are crucially important (here: editing actions vs. typing actions), and those in turn result from the roles embedded in the interaction design.

Finally, we provide examples for practical design decisions and discuss broader research implications. If model performance improves, we assume that generative models might replace tedious routine writing in the future. At the same time, they will aid human writing skills for creative and original text. In this light, our study highlights the importance to better understand the implications of designing human-AI interaction for writing creative texts with generative models. The overarching goal of our research is thus to contribute to future human-AI interaction that enhances human skill, instead of replacing it.

ACKNOWLEDGMENTS

This project is funded by the Bavarian State Ministry of Science and the Arts and coordinated by the Bavarian Research Institute for Digital Transformation (bidt).

REFERENCES

[1] Khaldoun Al Faraj, Mustapha Mojahid, and Nadine Vigouroux. 2009. BigKey: A Virtual Keyboard for Mobile Devices. In *Human-Computer Interaction. Ambient, Ubiquitous and Intelligent Interaction*, Julie A. Jacko (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 3–10. https://doi.org/10.1007/978-3-642-02580-8_1

[2] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. Institute of Electrical and Electronics

Engineers, Toronto, Ontario, Canada, 100–105. https://doi.org/10.1109/TIC-STH.2009.5444533

[3] Kenneth C. Arnold, Krysta Chauncey, and Krzysztof Z. Gajos. 2020. Predictive Text Encourages Predictable Writing. In *Proceedings of the 25th International Conference on Intelligent User Interfaces* (Cagliari, Italy) *(IUI '20)*. Association for Computing Machinery, New York, NY, USA, 128–138. https://doi.org/10.1145/3377325.3377523

[4] Kenneth C. Arnold, Krzysztof Z. Gajos, and Adam T. Kalai. 2016. On Suggesting Phrases vs. Predicting Words for Mobile Text Composition. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 603–608. https://doi.org/10.1145/2984511.2984584

[5] Kenneth C. Arnold, April M. Volzer, and Noah G. Madrid. 2021. Generative Models can Help Writers without Writing for Them. http://ceur-ws.org/Vol-2903/IUI21WS-HAIGEN-1.pdf

[6] Yoshua Bengio. 2008. Neural net language models. *Scholarpedia* 3, 1 (2008), 3881. https://doi.org/10.4249/scholarpedia.3881 revision #140963.

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. https://doi.org/10.48550/ARXIV.2005.14165 arXiv:2005.14165

[8] Daniel Buschek, Martin Zürn, and Malin Eiband. 2021. The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 732, 13 pages. https://doi.org/10.1145/3411764.3445372

[9] Alex Calderwood, Vivian Qiu, Katy Ilonka Gero, and Lydia B Chilton. 2020. How Novelists Use Generative Language Models: An Exploratory User Study. http://ceur-ws.org/Vol-2848/HAI-GEN-Paper-3.pdf

[10] Haw-Shiuan Chang, Jiaming Yuan, Mohit Iyyer, and Andrew McCallum. 2021. Changing the Mind of Transformers for Topically-Controllable Language Generation. https://doi.org/10.48550/ARXIV.2103.15335 arXiv:2103.15335

[11] Mia Xu Chen, Benjamin N. Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M. Dai, Zhifeng Chen, Timothy Sohn, and Yonghui Wu. 2019. Gmail Smart Compose: Real-Time Assisted Writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2287–2295. https://doi.org/10.1145/3292500.3330723

[12] Andy Coenen, Luke Davis, Daphne Ippolito, Emily Reif, and Ann Yuan. 2021. Wordcraft: a Human-AI Collaborative Editor for Story Writing. https://doi.org/10.48550/ARXIV.2107.07430 arXiv:2107.07430

[13] Wenzhe Cui, Suwen Zhu, Mingrui Ray Zhang, H. Andrew Schwartz, Jacob O. Wobbrock, and Xiaojun Bi. 2020. *JustCorrect: Intelligent Post Hoc Text Correction Techniques on Smartphones.* Association for Computing Machinery, New York, NY, USA, 487–499. https://doi.org/10.1145/3379337.3415857

[14] Lei Gao. 2005. Latin squares in experimental design. http://compneurosci.com/wiki/images/9/98/Latin_square_Method.pdf

[15] Surjya Ghosh, Kaustubh Hiware, Niloy Ganguly, Bivas Mitra, and Pradipta De. 2019. Does Emotion Influence the Use of Auto-Suggest during Smartphone Typing?. In *Proceedings of the 24th International Conference on Intelligent User Interfaces* (Marina del Ray, California) *(IUI '19)*. Association for Computing Machinery, New York, NY, USA, 144–149. https://doi.org/10.1145/3301275.3302329

[16] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language Modeling for Soft Keyboards. In *Proceedings of the 7th International Conference on Intelligent User Interfaces* (San Francisco, California, USA) *(IUI '02)*. Association for Computing Machinery, New York, NY, USA, 194–195. https://doi.org/10.1145/502716.502753

[17] Asela Gunawardana, Tim Paek, and Christopher Meek. 2010. Usability Guided Key-Target Resizing for Soft Keyboards. In *Proceedings of the 15th International Conference on Intelligent User Interfaces* (Hong Kong, China) *(IUI '10)*. Association for Computing Machinery, New York, NY, USA, 111–118. https://doi.org/10.1145/1719970.1719986

[18] Matthew Guzdial and Mark Riedl. 2019. An Interaction Framework for Studying Co-Creative AI. https://doi.org/10.48550/ARXIV.1903.09709 arXiv:1903.09709

[19] Eric Horvitz. 1999. Principles of Mixed-Initiative User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, USA) *(CHI '99)*. Association for Computing Machinery, New York, NY, USA, 159–166. https://doi.org/10.1145/302979.303030

[20] Torsten Hothorn, Kurt Hornik, Mark A. van de Wiel, and Achim Zeileis. 2008. Implementing a class of permutation tests: The coin package. *Journal of Statistical Software* 28, 8 (2008), 1–23. https://doi.org/10.18637/jss.v028.i08

[21] Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart Reply: Automated Response Suggestion for Email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 955–964. https://doi.org/10.1145/2939672.2939801

[22] Anna Kantosalo and Hannu Toivonen. 2016. Modes for Creative Human-Computer Collaboration: Alternating and Task-Divided Co-Creativity. In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*. Sony CSL, Paris, 77–84. http://www.computationalcreativity.net/iccc2016/wp-content/uploads/2016/01/Modes-for-Creative-Human-Computer-Collaboration.pdf

[23] Per Ola Kristensson and Keith Vertanen. 2014. The Inviscid Text Entry Rate and Its Application as a Grand Goal for Mobile Text Entry. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 335–338. https://doi.org/10.1145/2628363.2628405

[24] Alexandra Kuznetsova, Per B. Brockhoff, and Rune H. B. Christensen. 2017. lmerTest Package: Tests in Linear Mixed Effects Models. *Journal of Statistical Software* 82, 13 (2017), 1–26. https://doi.org/10.18637/jss.v082.i13

[25] Mina Lee, Percy Liang, and Qian Yang. 2022. CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 388, 19 pages. https://doi.org/10.1145/3491102.3502030

[26] Santiago Negrete-Yankelevich and Nora Morales-Zaragoza. 2014. The apprentice framework: planning and assessing creativity. In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2014)*. computationalcreativity.net, Ljubljana, 280–283. http://computationalcreativity.net/iccc2014/wp-content/uploads/2014/06/13.4_NegreteYankelevich.pdf

[27] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How Do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 9, 12 pages. https://doi.org/10.1145/3338286.3340120

[28] Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 83–88. https://doi.org/10.1145/2858036.2858305

[29] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9. http://www.persagen.com/files/misc/radford2019language.pdf

[30] Melissa Roemmele. 2021. Inspiration through Observation: Demonstrating the Influence of Automatically Generated Text on Creative Writing. https://doi.org/10.48550/ARXIV.2107.04007 arXiv:2107.04007

[31] Melissa Roemmele and Andrew S. Gordon. 2015. Creative Help: A Story Writing Assistant. In *Interactive Storytelling*, Henrik Schoenau-Fog, Luis Emilio Bruni, Sandy Louchart, and Sarune Baceviciute (Eds.). Springer International Publishing, Cham, 81–92.

[32] Nikhil Singh, Guillermo Bernal, Daria Savchenko, and Elena L. Glassman. 2022. Where to Hide a Stolen Elephant: Leaps in Creative Writing with Multimodal Machine Intelligence. https://doi.org/10.1145/3511599 Just Accepted.

[33] Paul D. Varcholik, Joseph J. LaViola, and Charles E. Hughes. 2012. Establishing a baseline for text entry for a multi-touch virtual keyboard. *International Journal of Human-Computer Studies* 70, 10 (2012), 657–672. https://doi.org/10.1016/j.ijhcs.2012.05.007 Special issue on Developing, Evaluating and Deploying Multi-touch Systems.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. https://doi.org/10.48550/ARXIV.1706.03762 arXiv:1706.03762

[35] Jacob O. Wobbrock and Matthew Kay. 2016. *Nonparametric Statistics in Human–Computer Interaction.* Springer International Publishing, Cham, 135–170. https://doi.org/10.1007/978-3-319-26633-6_7

[36] Mingrui Ray Zhang, He Wen, and Jacob O. Wobbrock. 2019. Type, Then Correct: Intelligent Text Correction Techniques for Mobile Text Entry Using Neural Networks. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 843–855. https://doi.org/10.1145/3332165.3347924

8 APPENDIX

8.1 Likert Questions

ID	Question	Response type
1	I am satisfied with the text.	5-point-scale (1=Strongly disagree, 5=Strongly agree)
2	It was easy for me to write the text.	
3	I feel like I am the author of the text.	
4	The interaction method was suitable for this task.	
5	The interaction method helped me write the text.	
6	The interaction method influenced the wording of the text.	

Table 1: Likert questions, we asked people after completing a task.

8.2 Questionnaire

ID	Question	Response type
A1	Enter your given user id.	Free-text
A2	What is your gender?	Selection (Woman, Man, Non-binary, Prefer not to disclose, Prefer to self-describe)
A3	What is your Age?	Number
A4	What is your English level (based on the CEFR Scale)?	Selection (C2, C1, B2, B1, A2, A1)
A5	Which is your dominant hand?	Selection (Right, Left, No dominant hand)
A6	How do you typically type on your mobile phone?	Selection (One handed with the thumb, One handed with the index finger, Two handed with both thumbs, Other)
A7	What kind of mobile phone did you use for the study?	Free-text
A8	Did you have any previous experience with writing with generated text? (Using suggestions etc. on mobile devices also counts.)	Selection (Yes, No)
A9	If yes, list all previous experiences which you had with writing with generated text.	Free-text

Table 2: Demographics

ID	Question	Response type
B1	Which of the two interaction methods, working with generated text, did you prefer for the task "It's your friend's birthday. Write an e-mail to wish all the best and mention that you have to meet again in some time."?	Selection (Continous generated text, Writing with suggestions)
B2	And shortly explain why you preferred this method for this task.	Free-text
B3	For this task ("It's your friend's birthday. Write an e-mail to wish all the best and mention that you have to meet again in some time."), would you prefer the standard input method over the method with generated text you chose before?	Selection (Yes, No)
B4	And can you shortly explain your decision?	Free-text
B5	Which of the two interaction methods, working with generated text, did you prefer for the task "Write a short story about your last or upcoming vacation."?	Selection (Continous generated text, Writing with suggestions)
B6	And shortly explain why you preferred this method for this task.	Free-text
B7	For this task ("Write a short story about your last or upcoming vacation."), would you prefer the standard input method over the method with generated text you chose before?	Selection (Yes, No)
B8	And can you shortly explain your decision?	Free-text

Table 3: Questions about the tasks

ID	Question	Response type
C1	How hard was it to learn and use this method?	5-point-scale (1=Very easy, 5=Very hard)
C2	How was the speed of the appearing text?	5-point-scale (1=Way too slow, 5=Way too fast)
C3	The interaction method inspired me to write things I normally wouldn't have thought of.	5-point-scale (1=Strongly disagree, 5=Strongly agree)
C4	I think the inspiration for new formulations / different wordings, provided by this interaction method, could be helpful in some cases. (Assuming the model would be better)	5-point-scale (1=Strongly disagree, 5=Strongly agree)
C5	In which situation/use-case could this interaction method be useful?	Free-Text
C6	What are the advantages of this method relating to the writing interaction?	Free-text
C7	What are the disadvantages of this method relating to the writing interaction?	Free-text
C8	Can you imagine any features which could make this method better?	Free-text

Table 4: Questions about "Continuous generated text"

ID	Question	Response type
D1	How hard was it to learn and use this method?	5-point-scale (1=Very easy, 5=Very hard)
D2	How was the number of suggestions provided?	5-point-scale (1=Way too low, 5=Way too high)
D3	How was the length of each suggestion?	5-point-scale (1=Way too short, 5=Way too long)
D4	The interaction method inspired me to write things I normally wouldn't have thought of.	5-point-scale (1=Strongly disagree, 5=Strongly agree)
D5	I think the inspiration for new formulations / different wordings, provided by this interaction method, could be helpful in some cases. (Assuming the model would be better)	5-point-scale (1=Strongly disagree, 5=Strongly agree)
D6	In which situation/use-case could this interaction method be useful?	Free-Text
D7	What are the advantages of this method relating to the writing interaction?	Free-text
D8	What are the disadvantages of this method relating to the writing interaction?	Free-text
D9	Can you imagine any features which could make this method better?	Free-text

Table 5: Questions about "Writing with suggestions"

Typing Behavior is About More than Speed: Users' Strategies for Choosing Word Suggestions Despite Slower Typing Rates

FLORIAN LEHMANN, Department of Computer Science, University of Bayreuth, Germany
ITTO KORNECKI, ETH Zurich, Switzerland
DANIEL BUSCHEK, Department of Computer Science, University of Bayreuth, Germany
ANNA MARIA FEIT, Saarland University, Saarland Informatics Campus, Germany

Mobile word suggestions can slow down typing, yet are still widely used. To investigate the apparent benefits beyond speed, we analyzed typing behavior of 15,162 users of mobile devices. Controlling for natural typing speed (a confounding factor not considered by prior work), we statistically show that slower typists use suggestions more often but are slowed down by doing so. To better understand how these typists leverage suggestions – if not to improve their speed – we extract eight usage strategies, including completion, correction, and next-word prediction. We find that word characteristics, such as length or frequency, along with the strategy, are predictive of whether a user will select a suggestion. We show how to operationalize our findings by building and evaluating a predictive model of suggestion selection. Such a model could be used to augment existing suggestion algorithms to consider people's strategic use of word predictions beyond speed and keystroke savings.

CCS Concepts: • **Human-centered computing → Empirical studies in HCI**; **Text input**; **Touch screens**.

Additional Key Words and Phrases: Text entry; word prediction; word suggestion; intelligent text entry methods; mobile text entry; typing

ACM Reference Format:
Florian Lehmann, Itto Kornecki, Daniel Buschek, and Anna Maria Feit. 2023. Typing Behavior is About More than Speed: Users' Strategies for Choosing Word Suggestions Despite Slower Typing Rates. *Proc. ACM Hum.-Comput. Interact.* 7, MHCI, Article 229 (September 2023), 26 pages. https://doi.org/10.1145/3604276

1 INTRODUCTION

Word suggestions are a widely used feature when typing on mobile devices such as smartphones. As the user types letters, the virtual keyboard suggests words that are predicted to match the intended input. Likely candidates are presented above the keyboard where the user can select them, as shown in Figure 1. This intelligent text entry (ITE) feature originates from augmentative and alternative communication (AAC) methods that support people with cognitive or physical impairments. It was developed to reduce the effort of communicating via text input. Instead of entering each letter of a word manually – which can take many seconds for AAC users – the complete word can be selected from the suggestion list, saving 30% to 50% of input actions [12, 19] and resulting in faster communication rates [13, 37].

However, for mobile users without any impairment, the use of word suggestions has been related to slower typing performance, despite keystroke savings. It comes at a cognitive cost for shifting

Authors' addresses: Florian Lehmann, florian.lehmann@uni-bayreuth.de, Department of Computer Science, University of Bayreuth, Bayreuth, Germany; Itto Kornecki, ittokor@gmail.com, ETH Zurich, Zurich, Switzerland; Daniel Buschek, daniel.buschek@uni-bayreuth.de, Department of Computer Science, University of Bayreuth, Bayreuth, Germany; Anna Maria Feit, feit@cs.uni-saarland.de, Saarland University, Saarland Informatics Campus, Saarbrücken, Germany.

2573-0142/2023/9-ART229
https://doi.org/10.1145/3604276

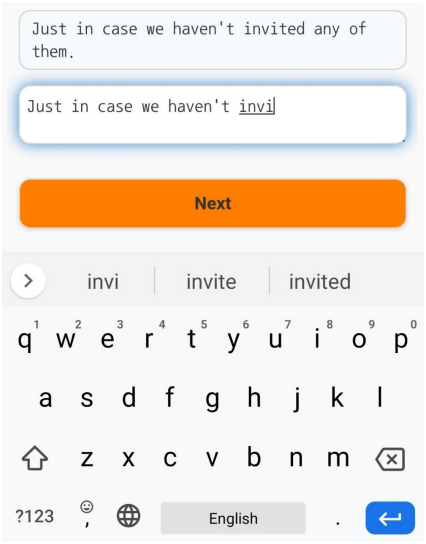


Fig. 1. Typical word suggestion interface on mobile devices. Screenshot from the typing test that provided the data for this work. The suggestion bar offers the currently typed string, and two word completions.

attention from pressing keys to reviewing and choosing words from the suggested list. This cost may outweigh the performance benefit [22, 25, 33]. Particularly on mobile devices, where users take only few hundred milliseconds to press keys [32], recent studies have observed that users of word suggestions were generally slower despite significant keystroke savings [3, 32, 33].

Nevertheless, word suggestions are a popular feature of mobile keyboards and studies reported that users perceive them to require less effort and are more satisfied with their typing experience [21, 33] despite the slower performance. Thus, we observe a discrepancy between text entry research and word suggestion usage in practice. While research on word suggestions is motivated by increasing typing speed, end-users utilize suggestions to improve their overall typing experience. How they integrate word suggestions into their typing process and the factors contributing to their typing experience are not well understood.

To address this, the goal of this paper is to better understand the way in which people make use of word suggestions during regular typing on mobile devices (not considering gesture typing), including – but going beyond – their effect on typing performance. Therefore, we analyze an existing dataset published by Palin et al. [32] which contains typing data from several thousand volunteers who participated in an online typing test. In contrast to prior work [3, 33], people were typing on their own devices and thus interacting with their familiar word suggestion algorithms. They were not paid to participate in the study but were intrinsically motivated coming to the webpage to learn about their mobile typing speed. This ensures a high external validity of the dataset and our findings here.

For the first time, we statistically show that typing speed and word suggestion usage mutually affect each other: Word suggestions are more commonly used by slower typists and at the same time suggestions slow down the typing process. To this end, we compute four different speed metrics that allow us to separate the motor processes and cognitive processes involved in typing with word suggestions. This enables us to assess the cognitive costs of suggestions and their effect on users' typing performance, disentangling the ambiguous correlations and average trends found by prior work [3, 32].

But how do people make use of word suggestions if not to improve their speed? We show evidence that users employ specific strategies when using suggestions, and that the strategy used depends heavily on the word which the user is typing. We identify eight strategies, including word completion and correction as the most prevalent ones. These significantly differ in how users type before making a selection and are particularly used for longer and infrequent words. In other cases, suggestions are strategically used to capitalize a word or to add an apostrophe for contraction words (e.g. *can't, don't*). In several cases, people also select incorrect predictions which are similar to their intended word, and then manually correct them to enter their intended word.

Our findings challenge the assumption often made by prior work, which is that users will select a word as long as it matches the one they are currently typing [12, 23, 25]. Instead, we show that typists strategically utilize suggestions for specific types of words. This indicates that suggestions are an integral part of a user's typing strategy and not simply a reaction to the keyboard displaying them. Crucially, this also implies that users plan ahead whether and how to use suggestions *before* these are displayed by the keyboard. This holds important implications for the design of ITE methods. To better support users, word suggestions should not only match the word a user is currently typing, but also consider how typists will want to make use of them. For example, instead of favoring frequent words which are more likely to be typed by the user, a suggestion algorithm could prioritize displaying less frequent words for which users are more likely to need a suggestion for (e.g. because they are unsure about spelling). We show how this could be achieved by deriving a selection model from our empirical findings that predicts whether a user will select a displayed suggestion based on the user's typing behavior and the characteristics of the word. We discuss how such a model could be used to augment existing algorithms to provide suggestions that are not just accurate, but also likely to be used by the typist. We make the code for our analysis and the selection model publicly available.

2 RELATED WORK

In the following, we briefly describe how word suggestions are computed on mobile keyboards. Although the underlying algorithms are not the focus of our work, they inform our analysis and findings. We then discuss prior work on the benefits and drawbacks of using word suggestions and show how such empirical findings have been used by researchers to inform the design of ITE systems.

2.1 Word suggestion algorithms

The foundation of most intelligent text entry methods is a statistical language model which predicts the probability of the next word or phrase given a series of words or characters the user has previously entered (see e.g. [30] for an overview). A typical approach uses n-gram dictionaries that contain the frequency with which various combinations of words occur in a text corpus. A word completion algorithm determines the most frequent words that start with the characters entered by the user so far. Modern keyboards on mobile devices combine such an algorithm with an error correction mechanism, a so-called decoder that infers a user's intended word from their noisy input actions (e.g. pressing small keys with a large thumb, resulting in entering the wrong characters). Therefore, a completion algorithm combines a language model with a touch model that estimates the probability that a series of observed touch points corresponds to a specific word [1, 16, 30, 38]. In recent years, much research has been dedicated to improving these language and touch models to accurately predict the word that the user intends to type next (e.g. [6, 15, 34, 40, 41]). In particular, recent advances in machine learning have also led to neural network-based language models for mobile word suggestion [18, 42]. While advanced models are able to capture the statistical relationships between a larger number of words to make more reliable predictions about the next

word a user wants to type, they all share the same assumption: if the correct word is suggested to the user, the user will choose it rather than typing the word manually. Little research has explored the tendency of typists to use these suggestions and the factors that influence these choices beyond the accuracy of the algorithm.

We show that even if a word is correctly suggested to the user, a user may still choose to type it manually. We explore how typists choose word suggestions as part of their typing process. We show that users strategically decide to use suggestions for specific types of words and, at times, they even choose wrong suggestions that are similar to their intended word. We propose a simple selection model which augments existing suggestion algorithms and takes into account user preferences.

2.2 Keystroke savings versus performance improvements

Today's word suggestions on mobile devices are largely inspired by early work in the 1980s and 1990s (e.g. [19, 22, 23, 29, 35]). These works aimed to improve augmented and alternative communication (AAC) devices which support people with cognitive or motor impairments to communicate via generated speech or written text.

Several studies have empirically and theoretically shown that word suggestion algorithms can yield large keystroke savings of at least 30 –50% [12, 14, 19, 33]. Given the amount of time that AAC users often require to type individual keys, it is reasonable to assume that such keystroke savings translate to faster typing speeds. Several studies have empirically confirmed this assumption [4, 29, 36]. At the same time, researchers recognized that the cognitive cost of using word suggestions might counteract the performance benefit from keystroke savings [22]. Choosing a word suggestion requires interrupting an often automated motor task, shifting attention to the suggested words, and deciding whether to choose one of them. There is a risk that the desired word might not be suggested, and switching attention from manual typing to the suggestion list might not pay off.

2.2.1 Keystroke savings do not translate to increased typing speed on mobile devices With the advent of mobile phones and touch interfaces, word suggestions became an integral part of standard virtual keyboards. However, some recent studies have observed that users type significantly slower when using word suggestions. Quinn and Zhai [33] found that using word suggestions reduced the typing speed from an average of 3.09 characters per second to 2.66 characters per second in a lab study and attributed this to the cognitive cost of evaluating and selecting the suggestions. The same was found by Arnold et al. [5] in the context of phrase suggestions, noting that the cost of accepting suggestions counteracts any speed benefit it may provide [5]. A large-scale online study of mobile typing by Palin et al. analyzed the effect of various ITE methods on typing speed. They found a negative correlation between users' typing speed and their number of chosen word suggestions. Similarly, in a crowd-sourcing study, Alharbi et al. [3] found that words chosen from the suggestion list were typed on average 2.09s slower than manually typed words. However, these studies did not control for the manual typing speed of participants, leaving open the question as to whether typing with word suggestions is indeed slower or whether the effect is caused by slower typists using word suggestions more often.

Overall, we can say that the performance benefits of word suggestion highly depend on the characteristics of the text entry method, the accuracy of the prediction algorithm, and the user's strategy for distributing their attention. We perform a detailed statistical analysis to disentangle the motor processes of typing from the cognitive processes involved in choosing suggestions and quantify the negative effect each chosen suggestion has on a user's typing performance, independent of their motor speed.

2.2.2 Typists objectives for using word suggestions Despite the apparent reduction in speed, word suggestions still benefit mobile users' by saving keystrokes [32, 33] and improving their subjective

typing experience. For example, Quinn and Zhai found that typists using word prediction perceived the typing experience to require less physical effort, and less effort overall, compared to typing normally [33]. Prior to this work, Kamvar and Baluja found a similar reduction in perceived workload when surveying typists using word prediction systems on 9-key mobile phones [21].

Beyond these few observations, prior studies have not considered the potential objectives of people to choose word suggestions despite their negative impact on performance. This paper takes a first step to explore users' strategic use of word suggestions that are motivated by other aspects than speed. We demonstrate that they are more likely to choose suggestions for specific types of words, such as capitalized words or contractions, purposefully omitting for example an apostrophe and relying on the algorithm to offer this suggestion.

2.3 Informing the design of ITE methods with user simulations

Simulation models have long been used in HCI to inform the design of interactive systems, a famous example being Card et al.'s Model Human Processor proposed in the 1980s [10]. Although their use is not as widespread as in other disciplines, the rise of advanced machine learning models and large-scale data also yields new computational approaches for HCI research [28]. In particular for text entry systems, researchers have long tried to model and simulate user performance for example to optimize keyboards (see e.g. [11] for an overview) but also to inform the design of ITE methods [12], in particular word suggestion algorithms.

For typing with word suggestions, Koester and Levine [23] and more recently Kristensson and Müller [25] developed simulation models that predict a user's typing performance as a result of their cognitive and motor characteristics, their interaction strategy, and the accuracy of the word suggestion algorithm. Such models can be used to assess the impact of the accuracy of the algorithm on overall performance or could evaluate design choices, such as varying the number of displayed suggestions as studied by [2, 7]) without having to run a user study.

These simulation models are based on empirical observations. Most studies were interested in understanding the effect the language model has on user performance. As such, they controlled and systematically varied the accuracy of the suggestion algorithm and observed user's performance [3, 33]. However, empirical studies that explore the interaction strategy of users when they select words from the suggestion list are missing. Thus, most simulation models assumed that users would choose a suggestion as soon as it is displayed [12] or formulated simple strategies where users check the suggestion list after a fixed number of characters [23, 25]. This reflects the general assumption that users will choose any accurate word suggestion. Recently, Oulasvirta, Jokinen, and Howes have proposed computational rationality as a theory for predicting interaction strategies as a result of the user adapting their behavior to an interface, their own cognitive and motor constraints, and their objectives and preferences [31]. While advances have been made to model the motor and cognitive characteristics of users typing on mobile keyboards [20], they do not include the use of word suggestions for which the objectives that guide a user's actions are unclear yet, as discussed above.

Thus, our research analyzes empirical data of word suggestion usage to identify common strategies which users employ when using word suggestions while typing. This will enable us to better understand people's objectives when typing, inform simulation models, and ultimately allow for better design of ITE methods.

3 DATASET

We build on an existing dataset by Palin et al. [32]. We decided on this dataset because of its large scale and its high external validity. We next describe the original data and our pre-processing steps to obtain a large-scale accurate dataset of word suggestion usage on mobile devices.

Actual keystrokes	Registered keyboard event
'T', 'h', 'e'	'T', 'Th', 'The' 'T', 'h', 'e', 'The' 'undefined', 'undefined', 'undefined'

Table 1. Types of faulty backend behavior which we filtered out to obtain a valid dataset of word predictions.

3.1 Original data: the 37K dataset

We obtained the dataset by Palin et al. [32] from their project page[1]. The authors collected mobile typing data from 37,370 volunteers who participated in an online typing test. People came to a commercial website to learn about their typing speed and voluntarily chose the typing experiment offered by the researchers among a list of standard tests provided by the webpage. In the typing test, participants transcribed 15 sentences randomly selected from the Enron mobile email corpus [39] and the English Gigaword Newswire corpus. They used their own keyboard and keystroke data was logged via the browser. The online typing test offers more control than an in-the-wild study and yields a larger sample of participants and mobile devices than a lab-based study. The intrinsic motivation of people to learn about their speed and the use of their own devices without any restrictions of modifications as done in other studies (e.g. [3] ensures realistic typing behavior and thus a high external validity of the collected data. The large scale of the dataset allows us to perform powerful statistical analyses. On the other side, people were self-selected, resulting in a dataset of rather young participants with a larger percentage of women. See the original paper for more details on the data collection [32].

3.2 Filtering

The browser-based logging comes with several limitations, such as restricted access to keypress information on some devices. Given the large size of the original dataset (over 37,000 users) we decided to rigorously exclude parts of the data to obtain accurate logs of word suggestion usage with a homogeneous quality.

We took three filtering steps:

(1) *Exclude users with invalid event data:* in particular for some Android devices, keystrokes were logged by the browser in unexpected ways. Table 1 shows several examples. We exclude those users to ensure consistent data logs, needed to reliably detect the selection of a suggested word (see below).
(2) *Exclude non-native speakers:* since the transcription task was in English, we exclude self-reported non-native speakers to increase the homogeneity and realism of the typing behavior.
(3) *Exclude swipe users:* gesture typing (or swiping) is a common technique where the user continuously draws from one letter to another to enter a word. Word suggestions are an inherent feature of this input method needed to disambiguate between similar gestures and to fix recognition errors. In contrast, manually entered text does not rely on the availability of suggestions. Thus, we expect swipe users to use suggestions differently than manual typists and remove self-reported swipe users and those for which we detected the use of gestures for more than 10% of words.

[1]https://userinterfaces.aalto.fi/typing37k/

3.3 After filtering: input data from 15,162 participants

After filtering, we arrived at data from 15,162 participants These participants reported their gender as female (10,565; 69.68%), male (3,762; 24.81%) or did not disclose (835; 5.51%). Their age ranged from 6 to 61 years, with a median of 21 years. This is similar to the original dataset [32]. In terms of operating systems, a large part of Android users was excluded due to invalid logging behavior yielding in 91.67% of participants using iOS.

3.4 ITE event detection

The web-based data collection is limited in what information it receives about each keystroke, thus the dataset does not include information about whether the user pressed a single key, chose a word suggestion or was automatically corrected by an algorithm. The original work solved this by classifying events based on the observable changes in the text input field. We decided to derive our own recognition scheme since our conservative filtering approach described above ensures that keystroke events have consistent metadata.

We label each keystroke as one of three events: user types normally (*none*), autocorrection is triggered (*autocorrect*), or a word is selected from the suggestion bar (*suggestion*). We do not consider the gesture class recognized by Palin et al. [32] since we previously filtered out all self-reported swipe users. We distinguish these events in three steps:

(1) *Based on input length:* Manually entered characters are recorded in the backend as a single-character string. Thus, all events with an input length of one are classified as *none*. All others are ITE events.
(2) *Based on edit distance:* We compute the Levenshtein distance between the current and previous state of the input field. Autocorrections have an edit distance of at least one character. Thus, all ITE events with an edit distance of 0 are labeled as *suggestion*, which might occur if a user selected a suggestion for a word they already typed out.
(3) *Based on inter-key interval (IKI)*: the remaining ITE events are distinguished based on the time between the current and previous input event. In the case of *autocorrection*, the event happens automatically after pressing the space key, thus yielding a small IKI value. Suggestion selection requires more time since it is considered to be cognitively more demanding [23]. After assessing the IKI distributions of the ITE events in our dataset, we define a double threshold to distinguish ITE events while minimizing false positives: ITE events with an IKI below 400ms, are classified as *autocorrection*, and those above 500ms are labeled as *suggestion*.

4 TYPING PERFORMANCE WITH WORD PREDICTIONS

In this part, we want to first explore how typing speed and word suggestion usage impact each other. In their original analysis of the dataset, Palin et al. observed a negative correlation between the number of suggestions used and participants' typing performance. However, this might depend on several interacting factors, including the accuracy of the algorithm, the users' cognitive abilities, and their motor skills [25]. To disentangle this correlation, we define four speed metrics that separate the motor and cognitive processes involved in typing. We then correlate these with the suggestion usage of typists. While it might not be sufficient to establish a causal relationship between typing speed and suggestion usage from our observational data, it gives us a more differentiated view on people's typing behavior on their own devices without restricting or affecting their normal interaction.

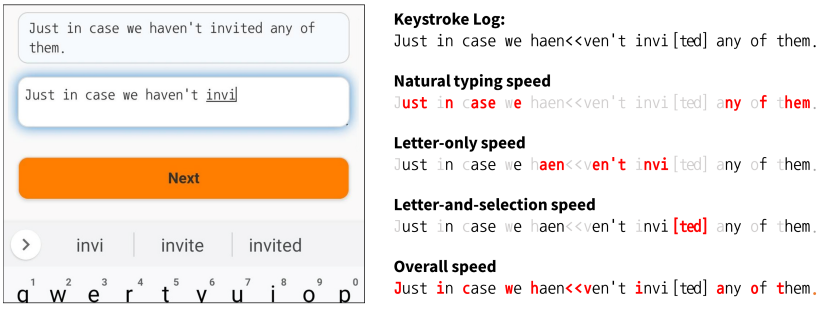


Fig. 2. We compute four different speed metrics to differentiate between motor expertise and the cognitive processes involved in typing, such as checking for and selecting suggestions. The first line shows an example keystroke log where a user corrected two mistakes (indicated by <) and entered the letters *ted* with a single keystroke (indicated by [ ]) by choosing a suggestion. The different metrics build on each other and are computed based on the highlighted keystrokes (in red and black). We use natural typing speed as a control variable when analyzing the impact of word suggestions on people's typing performance.

4.1 Performance metrics

We measured typing speed in four different ways, allowing us to assess the effect of choosing word suggestions on a user's overall typing performance. The intuition behind them is described in the following. Their computation is specified in Table 4 in the Appendix A.1 and exemplified in Figure 2. Each of them is measured in characters per second by averaging across the inter-key intervals of all keystrokes included in the computation. In addition, we measured keystroke savings as a standard performance metric used in the literature.

4.1.1 Natural speed We define the natural speed as a performance measure of the motor processes involved in pressing the keys on the keyboard. We thus *exclude* the use of word suggestions (or other ITE features) and only analyze keystrokes of lowercase letters that are preceded by lowercase letters, in common words shorter than five characters. In this way, we ensure to capture typing behavior that reflects a participant's motor skill, and minimize delays due to cognitive events such as checking the source text [24] or checking for suitable suggestions.

4.1.2 Letter-only speed The letter-only speed is measured similarly to the natural typing speed, except no restrictions are made regarding the typed word (e.g. including longer words). Thus, in addition to the motor speed, this measure includes the cognitive process of considering the use of suggestions and reviewing the word suggestion list, but it does not include the keyboard event where a suggestion is selected.

4.1.3 Letter-and-selection speed In addition to the previous keystrokes, this measure includes keyboard events that represent the selection of a suggestion. Thus, this speed involves the process of both reviewing and then selecting a word from the suggestion list.

4.1.4 Overall speed The overall speed involves all keystrokes, including non-letter ones, such as spaces, backspaces, corrections, capitalization, etc. It is equivalent to character-per-second speed evaluations made in the literature [27].

4.1.5 Keystroke savings In addition to typing speeds, we computed *keystroke savings* (KS) achieved by using word suggestions: We computed the keystrokes per character (KPC) by dividing the total

	Natural Speed	Letter-only Speed	Letter-and-selection Speed	Overall Speed
Non-suggestion users	6.13 (1.81)	5.6 (1.69)	5.6 (1.68)	4.19 (1.49)
Suggestion users	5.22 (1.43)	4.67 (1.28)	4.08 (1.21)	3.19 (0.86)
Difference[1]	–	-0.12*** (0.01)	-0.74*** (0.01)	-0.43*** (0.01)

[1] controlled for natural speed *** p<0.001

Table 2. Mean and standard deviation for speeds of suggestion and non-suggestion users. Difference was controlled for natural speed.

number of keystrokes by the total number of characters in the final text. We used the formula described by Garay-Vitoria and Abascal [13] to convert KPC to keystroke savings: $KS = 1 - \frac{1}{KPC}$

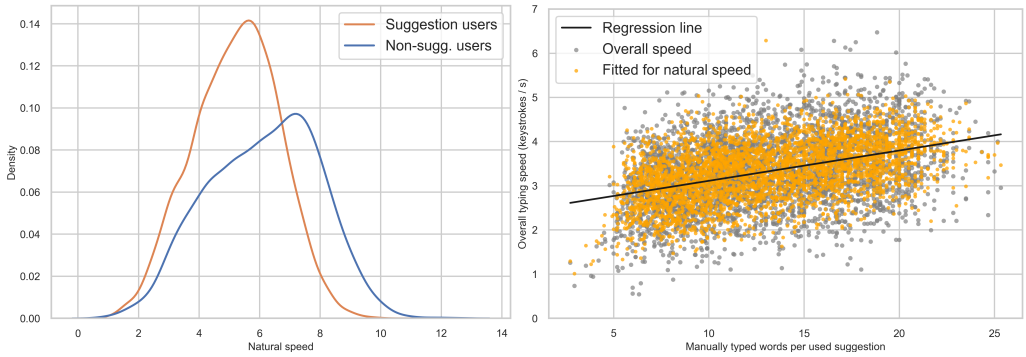
4.2 Suggestion- versus non-suggestion users

For analyzing the impact of suggestion usage on typing speed, we further filtered our dataset to only include people who never or frequently used word suggestions, yielding two user groups: (1) 3,926 *non-suggestion users* – who entered all characters manually and never chose a suggestion; and (2) 4,396 *suggestion users* – who chose suggestions more than five times during the 15 transcribed sentences. We first compared participants' natural typing speed, thus analyzing the differences in motor skill between *suggestion users* and *non-suggestion users*. We then compared the letter-only speed, letter-and-selection speed, and overall speed between the two user groups. All results are given in Table 2. The bottom row shows the performance differences between these two groups in keystrokes per second when controlling for their natural typing speed. Therefore, we performed ordinary least squares (OLS) analyses on each speed metric where the independent variables were users' natural speed and whether they used suggestions (a binary label). The results thus indicate how the usage of predictions affect typing speed at different levels of interaction, as described in the following.

Natural speed: The distributions of natural typing speeds for *non-suggestion users* and *suggestion users* as shown in Figure 3a descriptively show that there are more slower typists who make use of word suggestions. *Non-suggestion users'* natural speed was on average 6.10 keystrokes per second (SD = 1.81) versus 5.24 (SD = 1.43) for *suggestion users*.

Letter-only speed: Even when excluding the time taken to select a word from the suggestion list, *suggestion users* still type slower: They have a letter-only speed of 4.67 keystrokes per second (SD = 1.28), whereas *non-suggestion users* have 5.6 (SD = 1.69). Controlled for the difference in natural speed, *suggestion users* type 0.12 keystrokes per second slower (SD = 0.01). This could be attributed to the cognitive processes of checking the word suggestion list in cases where no suggestion is selected.

Letter-and-selection speed: Including the time taken to select words from the suggestion list, the difference in speed between *suggestion users* and *non-suggestion users* increases: The average letter-and-selection speed of *suggestion users* is 4.08 keystrokes per second (SD = 1.2) and 5.6 (SD = 1.68) of *non-suggestion users*. Controlled for difference in natural speed, *suggestion users* type 0.74 keystrokes per second slower (SD = 0.01). A large part of this difference could thus be attributed to the time it takes to decide for and select a suggestion from the word prediction list.



(a) (b)

Fig. 3. The effect of using word suggestion on typing speed. (a) Distributions of natural speed for *suggestion users* and *non-suggestion users*. (b) Effect of manually typed words per selected suggestion on typing speed: the more words are typed without suggestions, the faster the overall speed. Grey dots visualize the overall speed. Orange dots visualize the fitted values for overall speed, after controlling for natural speed.

Overall speed: Analyzing all the keystrokes, including also spaces, error corrections, and first letters of a word, show a smaller difference in typing speed: *Suggestion users* type 3.19 keystrokes per second (SD = 0.86) and *non-suggestion users* 4.19 (SD = 1.49). Controlled for natural speed, the difference is 0.43 keystrokes per second (SD = 0.01). Although there is a difference in typing speed that cannot be explained by the difference in natural speed, it is not as large as suggested by the difference in letter-and-selection speed. This could indicate word suggestions might be faster in certain cases, for example when correcting errors.

4.3 The impact of suggestion usage on typing performance

We further analyzed the effect of the rate at which suggestions were used. For this, we performed a multivariate regression using OLS on the data of *suggestion users*. We correlate the overall speed of users with their use of suggestions while controlling for their natural speed (i.e. natural speed and suggestion rate as independent variables, and overall speed as dependent variable). To ensure normality of the data, we compute suggestion rate as the number of *manually* typed words per used suggestion, e.g. a user might choose a suggestion every 10 words. The final analysis does not exclude outliers since it did not change the results. We refer to our analysis scripts for further assessments on the linearity and homoscedasticity of the data.

The results are visualized in Figure 3b and show that the more words are typed without a suggestion, the faster the user types: For every additional manually typed word per suggested word, the overall speed increased by 0.029 keystrokes per second (SD = 0.002, $p < 0.001$). This analysis takes into account the natural typing speed of users, that means, for two participants with the same natural speed, a user selecting a suggestion once in ten words will type 0.42 keystrokes per second slower than a user selecting a suggestion only once in 20 words. Comparing the gray and orange dots in Figure 3b, we see that controlling for participants' natural typing speed reduces parts of the variation in the data and shows a clear linear relationship between suggestion usage and overall speed after controlling for users' motor skills. However, we also see note that there is considerable variance left which cannot be explained by natural typing speed or suggestion usage.

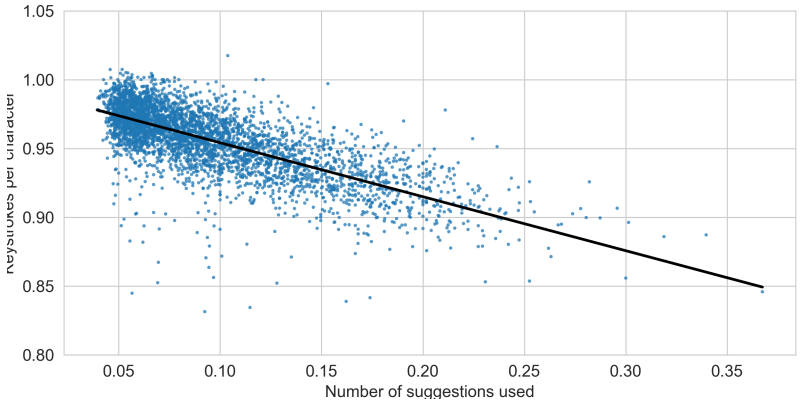


Fig. 4. Incremental keystrokes savings of suggestion usage.

4.4 Keystroke savings

We examined the effect of using suggestions on KPC and keystroke savings using an OLS analysis where the rate of suggestion usage (in terms of percentage of words for which suggestions were selected) was the independent variable, and the keystrokes per character were the dependent variable. Similar to above, we did not exclude outliers.

The result is visualized in Figure 4 (r^2 = 0.484). The results show that increased usage of suggestions leads to a decrease in the number of keystrokes per character (KPC). Our analysis shows for every 1% increase in the usage rate of suggestions, the KPC decreases by 0.004 (SD = 6E-5, p < 0.001). Practically speaking, a user selecting a suggestion once every 10 words will have keystroke savings of 4%, while a user selecting a suggestion once every five words will have keystroke savings of 8%.

4.5 Summary

By controlling for natural typing speed, we have found that there is a relationship between typing speed and the use of word suggestions: Typists who type more slowly tend to rely more often on suggestions. Conversely, the use of suggestions also slows down the typing speed. We see a difference between *suggestion users* and *non-suggestions users* even if we exclude the time it takes to choose a suggestion from the suggestion list. Thus, the keystroke savings achieved through the use of word suggestions do not translate to performance improvements in mobile typing. We further discuss these results in the Discussion section.

5 USERS' STRATEGIES FOR SELECTING WORD SUGGESTIONS

To better understand how and why people use suggestions, if not for speed, we looked for trends in which words users tend to select a suggestion vs typing them manually. We refer to these trends as *strategies* throughout this paper.

Without prior ground truth and a vast feature space associated with the words and typing behavior, it is extremely difficult to identify strategies using unsupervised methods (e.g. clustering). Therefore, we employed a semi-qualitative approach: In the first step, we combined insights from manual inspection of the typing data with our experience and domain knowledge about word suggestion use. Based on these insights, we formed hypotheses about potential strategies. In the second step, we implemented a rule-based classifier to identify all occurrences of these strategies in the dataset. We iterated both steps to narrow down a list of frequent strategies.

Next, we first describe the identified strategies, before taking a closer look at the two most common ones.

5.1 Selection strategies

Table 3 provides an overview of the eight strategies we found. Their associated rules for labeling them in the data can be found in Table 5 in the Appendix. In total, the eight strategies account for 98.5% of all suggestion usage. The strategy *completion* is used most often with 60.6%, followed by *correction* with 28.3%. Together, these two account for almost 90% of all suggestion usage. All other strategies occur with less than 5% individually.

Note that the frequency of a strategy depends heavily on the words it is applied on. For example, the *contraction* strategy is only used for 2.6% of all suggestion usage but accounts for 48.9% of suggestion usage of contraction words. Similarly, *capitalization* accounts for 23.6% for suggestions of capitalized words (vs. 2.5% overall).

Strategy	Description	Frequency
Completion	The user types the beginning of the word and then selects a word from the suggestion list to complete it. E.g. user types "wat" and selects "watch".	60.60%
Correction	The user types part or all of a word and then selects a suggestion to correct a mistake that was made. E.g. user types "abliity" and selects "ability".	28.30%
Prediction	The user selects a word from the suggestion list before typing anything.	3.70%
Contraction	The user types a contraction word fully without the apostrophe and then selects a word from the suggestion list to add it. E.g. user types "cant" and selects "can't".	2.60%
Capitalization	The user types a capitalized word fully in lower case, and then selects a word from the suggestion list to add capitalization. E.g. user types "bob" and selects "Bob".	2.50%
Insert space	The user fully types two or more words but forgets to add a space between them. The user then uses the suggestions to add a space between the words. E.g. User types "thereis" and selects "there is".	0.40%
Fixup	The user makes a mistake while typing a word. To correct the mistake, the user partially erases the word and selects a new word from the suggestion list. E.g. user types "press", then erases to get "pres", and then selects "president".	0.40%
Select-and-modify	A "meta-strategy": The user selects a suggestion (potentially with any other strategy) and modifies it. We found these types: Adding or removing characters at the end (e.g. select "apples", modify to "apple") and replacements at the end (e.g. "responsible" to "responsibility").	5.0%

Table 3. Description and frequency of the eight strategies for using word suggestions we found in our analysis.

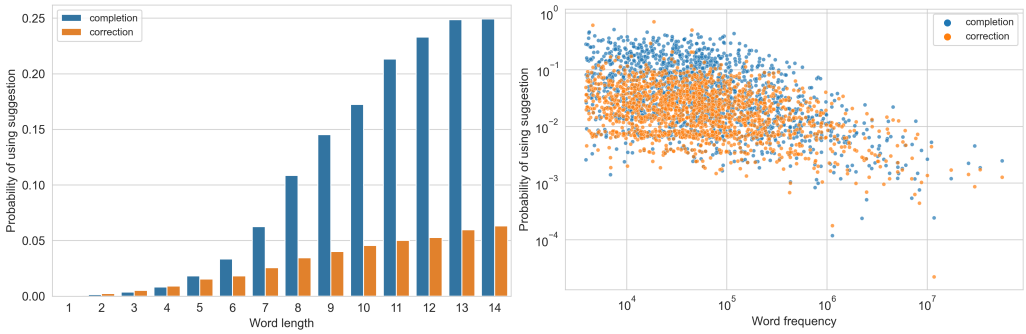


Fig. 5. Effect of word characteristics on the probability of using word suggestion. (a) Word length. (b) Word frequency.

5.2 A closer look at completion and correction strategies

As *completion* and *correction* are the most used strategies, we take a closer look at them here: In particular, we analyzed the effects of word length and frequency, and examine the related typing behavior.

5.2.1 Effect of word length Figure 5a descriptively shows the effect of word length (in characters) on these two strategies. Longer words have a higher probability of being used as part of the two strategies. We verify this by examining length vs frequency: For words with four characters, *correction* is only used for 0.90% of all words and *completion* for 0.82%. In contrast, for words with 14 characters, *correction* is used for 6.3% and completion for 25%. These results indicate that for both strategies, longer words are more frequently entered with the use of suggestions.

5.2.2 Effect of word frequency Figure 5b shows the effect of word frequency (as per [17]) on suggestion usage. The probability of using suggestions in the *completion* or *correction* strategy decreases for more frequently used words. For further comparison, we categorized words into common words (frequency > 100,000 in [17]) and uncommon ones (<100,000): For suggestions of common words, *completion* was used 1.8% and *correction* 0.42%. For uncommon words, *completion* and *correction* were used 5.5% and 2.7% of the time, respectively. These results indicate that getting support for entering uncommon words is a motivator for using word suggestions.

5.2.3 Typing speed before selecting suggestions We examined the *letters-only typing speed* (defined in Section 4.1) for the characters of a word that were typed before selecting a suggestion for that word. Figure 6 visualizes this *lead-up speed*. In summary, the lead-up speed for words without suggestion usage was 5.8 keystrokes per second (SD = 2.13). If the *completion* strategy was used for a word, that speed was 3.46 (SD = 1.70), and for the *correction* strategy, it was 4.63 (SD = 2.04). An ANOVA revealed a significant difference between the speeds of these strategies ($p < 0.001$). This indicates that the negative performance impact of using suggestions varies between strategies. This could be explained by the fact that the completion strategy might involve checking the word prediction list more frequently, whereas for correction it might only be checked at the end of a word, once a typing mistake was detected.

5.2.4 Edit distance We analyzed the edit distance between the manually typed word and selected suggested word. For *completion* (Figure 7a), this edit distance increases with word length. For *correction* (Figure 7b), the distance stays close to 1 regardless of word length. For a word of four

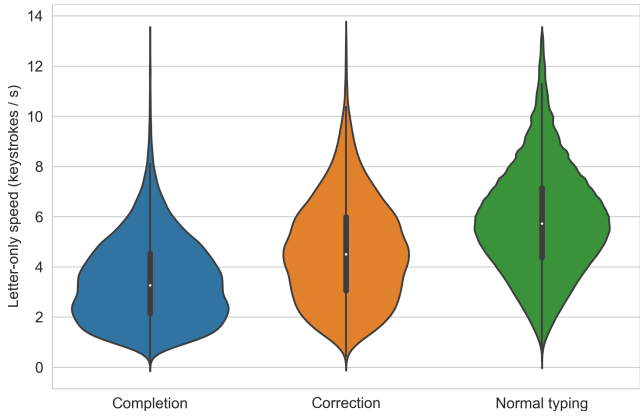
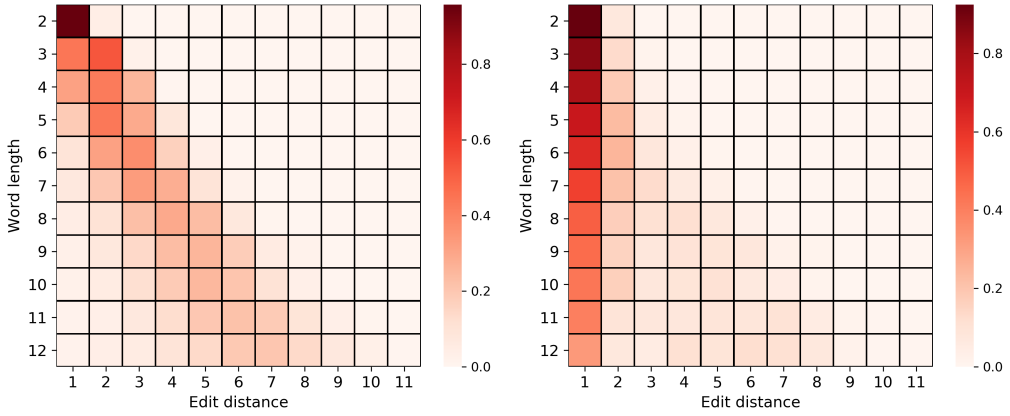


Fig. 6. Typing speed leading up to a selection for the strategies of *completion* and *correction*. For comparison, we show typing speed for words where suggestion is not used.



(a) Completion (b) Correction

Fig. 7. Edit distance of completion and correction for increasing word lengths, per strategy. Color indicates frequency.

characters, for example, *completion* and *correction* result in a median edit distance of two and one character(s), respectively. For a word of 10 characters, the median edit distance for *completion* is 5 characters and for *correction* 2 characters. These results further support the bigger picture: The strategies differ in various aspects. Concretely, in addition to word length, frequency, and speed, this also reflects in how much the suggestion changes/extends the typed (parts of a) word.

5.3 Summary

In summary, we identified eight strategies for how typists use word suggestions. The strategies *completion* and *correction* are used most often and users particularly prefer to use them for long and uncommon words.

Furthermore, they used suggestions strategically to modify words, such as capitalization or adding apostrophes for contractions (e.g. *can't* or *don't*), which was used for almost 50% of contraction

words. Another approach of strategically using suggestions happened in parallel to the main strategies: Users select a prediction close to the intended word to modify it afterwards.

6 APPLICATION: IMPROVING SUGGESTION ALGORITHMS

Our analysis shows that the characteristics of a word influence whether a user will select a corresponding word suggestion. Moreover, we saw that their typing behavior changes depending on the strategy the suggestion corresponds to (e.g. completion versus correction). This has important implications for the design of word suggestion algorithms, which are typically optimized to show accurate suggestions, not considering whether users will choose them.

In this section, we show how our empirical findings could be operationalized to improve existing word suggestion algorithms. For demonstrational purposes, we developed a logistic regression model that predicts the probability that a displayed (correct) suggestion would be selected by a user. In the following, we give a brief intuition on how this selection model works and how it could be combined with a standard suggestion algorithm. In the Appendix, we provide detailed information on the logistic regression model we built from our dataset, including the features' coefficients, their statistical significance, and an assessment of its performance.

For computing the probability that the user will select a suggested word, we introduce the selection model P_S. This model predicts a probability based on the input behavior of the user, the characteristics of the word, and the corresponding strategy. Mathematically, this can be formulated as:

$$P_S = P(selection|i, w, strategy) \tag{1}$$

Where i is the set of features representing the current input of the user, w are the characteristics of the suggested word (e.g. the length) and $strategy$ is the strategy that would be invoked by selecting the word from the suggestion list. Typing "wat" and selecting "watch" from the suggestion list, for example, implies a completion strategy.

The predictions of the selection model could be directly combined with the output of a suggestion algorithm to make accurate predictions that will also be chosen by users. This can be formulated as follows:

$$P = P_A \times P_S \tag{2}$$

Where P_A is the word probability estimated by an existing keyboard prediction algorithm (e.g. with language and touch models) and P_S is the probability estimated by the selection model, as described above.

With a practical example, we describe the process of how the model can be used to estimate the probability of selections and rerank candidates in the suggestion list. We suppose for this example, the user has the intent to write the phrase *I love musicals*, and already typed the beginning *I love m*.

(1) The user types the letter *u*. The current string is *I love mu*.
(2) The language model and the touch model suggest several words that could likely extend the sentence. We give three examples and their hypothetical probability score P_A: *my*=0.5, *music*=0.3, and *musicals*=0.08.
(3) For each suggestion, the strategy is inferred (see Appendix): *my* is a correction, *music* a completion, *musicals* a completion.
(4) For each candidate, the selection model computes P_S, which is the likelihood of the word being selected if suggested to the user. For example, *my*=0.01 (e.g. since the word is very short), *music*=0.04, and *musicals*=0.2 (e.g. since it is longer and infrequent).

(5) Probabilities of the language model are combined with the probability of the selection model by multiplication, yielding: *my*=0.005, *music*=0.012, *musicals*=0.016.
(6) In a typical interface (see Figure 1), the top two suggestions are then presented to the user ordered by descending score, together with their current input.
(7) Repeat steps 2 to 7, every time a user inserts a new character.

This process illustrates how the selection model could be used to influence the order of suggestions to not only focus on the accuracy of the suggestion but also consider whether the suggestion will be useful to the user.

7 DISCUSSION

Our results show that there are more slower typists who make use of word suggestions on mobile keyboards. At the same time, we find suggestion users typed slower than people that do not use suggestions. This difference was found to be significant even when controlling for participants' natural typing speed. These typists make use of suggestions willing to suffer potential speed penalties through their use of word suggestions. With our in-depth analysis of their suggestion usage, we identify specific strategies that describe how typists make use of word suggestions and integrate them into their typing process. Our empirical findings could inform a predictive model of suggestion usage and we showed how such a model could be used to refine existing suggestion algorithms. In the following, we discuss our findings on decreased typing performance, the identified strategies, and our selection model.

7.1 There are more slower typists who make use of word suggestions, but by doing so they type slower

We found that word suggestions decrease typing performance on mobile devices. This confirms observations from prior studies [3, 32, 33]. However, our speed analysis controls for a confounding factor not considered by prior work: natural typing speed. This allowed us to quantify the effect the suggestion usage rate has on a person's typing performance. As a result, our work contributes to the literature with the insight that word suggestions and typing speed are related in *two* directions: 1) there are more slower typists that use word suggestions in the first place, but 2) by doing so they type even slower.

We discuss two possible explanations: Either, slow typists are aware of their slow typing and therefore use suggestions in an effort to increase their typing speed (and fail to do so). Or, they are less concerned about speed and use suggestions to reduce typing effort or with some other objective in mind. The former motivates a discussion of the costs of using suggestions, the latter a closer look at the identified strategies. We discuss both aspects in the following sections (Section 7.2, Section 7.3)

7.2 Cognitive costs outweigh speed benefits

Why do suggestions decrease speed? Our results indicate two reasons: First, suggestions introduce cognitive costs, as users have to read the shown suggestions and check for a word that fits their intention. This is in line with related work, which reports increased costs for each additional word in a suggestion list [8, 13, 23].

Second – and beyond this literature – we found that suggestions reduced the speed of *manual keystrokes*: Suggestion users typed more slowly overall, even if we excluded the selection times and controlled for natural speed. A possible explanation is that the mere presence of suggestions adds cognitive costs. This might be caused by users thinking about how and when to use suggestions

while typing, or by users switching attention and gaze between the suggestion list and the keys of the keyboard.

This finding connects to theoretical observations by Kristensson and Müllners [25] on the goodness of a typing strategy using word predictions. They characterize the trade-off between the physical cost of typing and the cognitive cost of using word predictions, as well as the prediction algorithm. Here, we add two empirical insights: First, we empirically show how this trade-off plays out in practice across many people and their keyboards and suggestion algorithms. In particular, suggestions decreased performance but are still used. This leads to the second insight, namely that a closer look is needed at the underlying behaviour: If not for speed, which objectives motivate users to use suggestions? Our analyses concretely identified eight strategies, which extend the literature on typing with suggestions, as discussed in detail next.

7.3 Typists strategically use word suggestions as a "toolbar"

Our results challenge a dominant assumption in prior work on word predictions: Users do not simply select a word if it is close to their intended word. Instead, they use the list of suggestions as a kind of "shortcut toolbar" strategically checking it for specific words and ignoring it otherwise. For example, when applying the strategies of *contraction* and *capitalization*, users employ suggestions as shortcuts to avoid manually typing apostrophes and capitalization.

These and other strategies also highlight the importance of considering a much broader range of motivations when studying word suggestions and designing user interfaces that offer them: Users strategically choose suggestions depending on (perceived) properties of the word they type, not only to avoid typing it at all in the first place. Concretely, these are properties that make words cumbersome to type, for instance, because of their length, familiarity, or required operations on mobile keyboards (e.g. switching mode to CAPS or special characters). Such observations can be interpreted in terms of the theory of computational rationality: users adapt their typing behaviour to internal (motivation, skills) and external bounds (word and keyboard characteristics) [20, 31].

The most frequently occurring strategies in our analysis are *completion* and *correction*. Note that this contradicts the core premise of today's large language models: Such models generally assign higher scores to more *common* words (in a given context), as learned from a training corpus. However, as we show, *completion* and *correction* are mainly used as shortcuts for long and *uncommon* words. Therefore, our results suggest that current language models could be improved for the specific use case of word prediction in keyboards, for example, by adding a new cost term in model training that reflects the behaviour strategies and features identified here.

7.4 Limitations and future work

Since we relied on existing data from a transcription task [32] our list of strategies might exclude some strategies or bias their frequencies in our analysis. The self-selected sample of participants includes younger and more female users who took the test to learn about their typing speed. People with more diverse cognitive or motor abilities, in particular older people are not well represented in the dataset and might differ in their strategic use of predictions. The transcription task might have influenced users to check the presented text to resolve spelling uncertainties instead of using suggestions. Another strategy that is excluded because of the transcription task is the use of suggestions for inspiration. For other tasks, such as writing emails, short messages, or notes, the list of strategies and their overall usage might differ. To explore more strategies, we suggest to investigate these use cases in future work. Lab studies building on our findings here could also involve eye tracking, as gaze behavior can be expected to vary between strategies as well.

We focus on the analysis of word suggestion usage with default mobile keyboards of Android and iOS. However, many keyboards offer further writing support features, such as auto-completion

and auto-correction or suggest whole phrases. Such features might affect the strategies or afford new ones. For example, we expect decreased use of *completion* and *correction* when typing with features that change/extend words automatically. Future work could study the potentially more complex strategies introduced by such features, for example, the likely pattern of correcting faulty or unwanted auto-completions and auto-corrections [3].

We have presented a simple yet interpretable logistic regression model to explore which features contribute to the selection of a word, and as a demonstration towards turning our analytical findings into actionable improvements for keyboards. This opens a space for future work to improve on our baseline model for applications in keyboards, for example, by investigating more features, training on different datasets, and exploring other models and training methods (e.g. Deep Learning). With our model and its evaluation metrics reported here, we provide a useful baseline for comparison.

7.5 Designing intelligent text entry systems for more than performance

In line with the literature, our analysis focused on input behaviour, yet what we found encourages a broader scope: Our findings imply that text suggestion systems could be improved on both the UI and algorithm level, considering the identified user strategies and underlying user motivations beyond speed. In this light, we argue that the perspective on suggestions in the text entry research community could be extended even further: Since we now know that speed is not the users' only motivation for using intelligent keyboard features, we argue that neither should it have to remain such a dominant goal for designers and developers.

What else might be considered? To give one example, future text suggestion systems could be mindful of factors of *agency*. Recent work has found effects of the UI design of suggestions on agency-related factors, such as perceived control and authorship when writing with phrase suggestions on a smartphone [26].

This broader perspective opens up fresh design directions, informed by our analyses here: For instance, mobile keyboard UIs could be designed to support the strategies more directly, adapt to them, or be open to adaptation by users. As a simple, concrete example, instead of a binary on/off switch for suggestions, keyboard apps could offer toggles for different strategies (e.g. suggest capitalization? suggest contractions? etc.). Furthermore, algorithms could be designed to show text that is informed not only by word frequency, but also by strategic value, or even agency-related values. For instance, a keyboard might suggest typical phrases in a functional business email but only suggest single words in a chat with a close friend or partner where "personal" authorship might matter more to the writer and receiver.

8 CONCLUSION

We have analyzed a large-scale realistic dataset of typing with and without word suggestions, leading to two key insights:

First, we have confirmed and enriched the emerging picture in the literature about the negative impact of word suggestions on typing speed: Concretely, controlling for natural typing speed enabled us to show that there are more people who type slowly to begin with who make use of word suggestions. They are then further slowed down by doing so.

This raises the question of why and how people make use of word suggestions, if not to improve their speed, leading to our second insight: Users employ specific strategies when using suggestions. Concretely, we identified eight such strategies here, including using suggestions for completion, correction, capitalization, and contraction.

In conclusion, a main takeaway from our analysis for the community is that the suggestion list in the UI is used more like a "toolbar" – a perspective which we argue may open up powerful new design directions. Furthermore, this also has implications for algorithms and models: For example,

current language models generally assign higher scores to more common words but users employ strategies that desire suggestions as shortcuts for uncommon or cumbersome to type words.

Overall, our findings thus encourage further research with a much broader perspective on user motivations and needs for mobile text suggestions. We hope our empirical analysis and its discussion encourage future research and design around intelligent text entry systems to look beyond speed.

We make our processed data and analysis scripts available, https://osf.io/u9aej/.

Acknowledgments

This project is funded by the Bavarian State Ministry of Science and the Arts and coordinated by the Bavarian Research Institute for Digital Transformation (bidt). Part of this work was done while the last author was a researcher at ETH Zurich.

References

[1] 2022. *Statistical Keyboard Decoding*. Cambridge University Press, 188–211. https://doi.org/10.1017/9781108874830.009

[2] Jiban Adhikary, Max Isom, and Keith Vertanen. 2022. The Impact of Number of Predictions on User Performance in a Dwell Keyboard. In *MobileHCI 2022 Workshop on Shaping Text Entry Research for 2030.* event-place: Vancouver, BC.

[3] Ohoud Alharbi, Wolfgang Stuerzlinger, and Felix Putze. 2020. The Effects of Predictive Features of Mobile Keyboards on Text Entry Speed and Errors. *Proceedings of the ACM on Human-Computer Interaction* 4, ISS (Nov. 2020), 1–16. https://doi.org/10.1145/3427311

[4] Denis Anson, Penni Moist, Mary Przywara, Heather Wells, Heather Saylor, and Hantz Maxime. 2006. The Effects of Word Completion and Word Prediction on Typing Rates Using On-Screen Keyboards. *Assistive Technology* 18, 2 (Sept. 2006), 146–154. https://doi.org/10.1080/10400435.2006.10131913

[5] Kenneth C. Arnold, Krzysztof Z. Gajos, and Adam T. Kalai. 2016. On Suggesting Phrases vs. Predicting Words for Mobile Text Composition. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. Association for Computing Machinery, Tokyo, Japan, 603–608. https://doi.org/10.1145/2984511.2984584

[6] Daniel Buschek and Florian Alt. 2015. TouchML: A Machine Learning Toolkit for Modelling Spatial Touch Targeting Behaviour. In *Proceedings of the 20th International Conference on Intelligent User Interfaces* (Atlanta, Georgia, USA) *(IUI '15)*. Association for Computing Machinery, New York, NY, USA, 110–114. https://doi.org/10.1145/2678025.2701381

[7] Daniel Buschek, Benjamin Bisinger, and Florian Alt. 2018. ResearchIME: A Mobile Keyboard Application for Studying Free Typing Behaviour in the Wild. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–14. https://doi.org/10.1145/3173574.3173829

[8] Daniel Buschek, Martin Zürn, and Malin Eiband. 2021. The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 732, 13 pages. https://doi.org/10.1145/3411764.3445372

[9] Leonard S. Cahen, Marlys J. Craun, and Susan K. Johnson. 1971. Spelling Difficulty. A Survey of the Research. *Review of Educational Research* 41, 4 (Oct. 1971), 281. https://doi.org/10.2307/1169440

[10] Stuart K. Card, Thomas P. Moran, and Allen Newell. 1983. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum. https://books.google.de/books?hl=de&lr=&id=2EoPEAAAQBAJ&oi=fnd&pg=PP1&ots=DSSP2kUOhL&sig=lQMAhA40COrxLlnTllkKlcuY8rE&redir_esc=y#v=onepage&q&f=false

[11] Anna Maria Feit. 2018. Assignment Problems for Optimizing Text Input. , 182 + app. 56 pages. http://urn.fi/URN:ISBN:978-952-60-8016-1

[12] Andrew Fowler, Kurt Partridge, Ciprian Chelba, Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2015. Effects of Language Modeling and Its Personalization on Touchscreen Typing Performance. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 649–658. https://doi.org/10.1145/2702123.2702503

[13] Nestor Garay-Vitoria and Julio Abascal. 2006. Text prediction systems: a survey. *Universal Access in the Information Society* 4, 3 (March 2006), 188–203. https://doi.org/10.1007/s10209-005-0005-9

[14] Nestor Garay-Vitoria and Julio González-Abascal. 1997. Intelligent word-prediction to enhance text input rate (a syntactic analysis-based word-prediction aid for people with severe motor and speech disability). In *Proceedings of the 2nd international conference on Intelligent user interfaces - IUI '97*. ACM Press, Orlando, Florida, United States, 241–244. https://doi.org/10.1145/238218.238333

[15] Mayank Goel, Leah Findlater, and Jacob Wobbrock. 2012. WalkType: using accelerometer data to accomodate situational impairments in mobile touch screen text entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Austin Texas USA, 2687–2696. https://doi.org/10.1145/2207676.2208662

[16] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language modeling for soft keyboards. In *Proceedings of the 7th international conference on Intelligent user interfaces - IUI '02*. ACM Press, San Francisco, California, USA, 194. https://doi.org/10.1145/502716.502753

[17] Project Gutenberg. 2006. Wiktionary:Frequency lists/PG/2006/04/1-10000. https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists/PG/2006/04/1-10000 last accessed: 2023-01-25.

[18] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated Learning for Mobile Keyboard Prediction. https://doi.org/10.48550/ARXIV.1811.03604

[19] D. Jeffery Higginbotham. 1992. Evaluation of keystroke savings across five assistive communication technologies. *Augmentative and Alternative Communication* 8, 4 (Jan. 1992), 258–272. https://doi.org/10.1080/07434619212331276303

[20] Jussi Jokinen, Aditya Acharya, Mohammad Uzair, Xinhui Jiang, and Antti Oulasvirta. 2021. Touchscreen Typing As Optimal Supervisory Control. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 720, 14 pages. https://doi.org/10.1145/3411764.3445483

[21] Maryam Kamvar and Shumeet Baluja. 2008. Query suggestions for mobile search: understanding usage patterns. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*. ACM Press, Florence, Italy, 1013. https://doi.org/10.1145/1357054.1357210

[22] Heidi Horstmann Koester and Simon Levine. 1996. Effect of a word prediction feature on user performance. *Augmentative and Alternative Communication* 12, 3 (Jan. 1996), 155–168. https://doi.org/10.1080/07434619612331277608

[23] Heidi Horstmann Koester and Simon Levine. 1998. Model simulations of user performance with word prediction. *Augmentative and Alternative Communication* 14, 1 (Jan. 1998), 25–36. https://doi.org/10.1080/07434619812331278176

[24] Andreas Komninos, Angeliki Tsiouma, Georgia Gogoulou, and John Garofalakis. 2022. Don't Look Up: The Cost of Attention to Stimulus Phrases in Mobile Text Entry Evaluations. *Pan-Hellenic Conference on Informatics*. https://doi.org/10.1145/3575879.3576015

[25] Per Ola Kristensson and Thomas Müllners. 2021. Design and Analysis of Intelligent Text Entry Systems with Function Structure Models and Envelope Analysis. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 584, 12 pages. https://doi.org/10.1145/3411764.3445566

[26] Florian Lehmann, Niklas Markert, Hai Dang, and Daniel Buschek. 2022. Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship. In *Proceedings of Mensch Und Computer 2022* (Darmstadt, Germany) *(MuC '22)*. Association for Computing Machinery, New York, NY, USA, 192–208. https://doi.org/10.1145/3543758.3543947

[27] I. Scott MacKenzie and R. William Soukoreff. 2002. Text Entry for Mobile Computing: Models and Methods,Theory and Practice. *Human–Computer Interaction* 17, 2-3 (2002), 147–198. https://doi.org/10.1080/07370024.2002.9667313

[28] Roderick Murray-Smith, Antti Oulasvirta, Andrew Howes, Jörg Müller, Aleksi Ikkala, Miroslav Bachinski, Arthur Fleig, Florian Fischer, and Markus Klar. 2022. What Simulation Can Do for HCI Research. *Interactions* 29, 6 (nov 2022), 48–53. https://doi.org/10.1145/3564038

[29] Alan F. Newell, Lynda Booth, John Arnott, and William Beattie. 1992. Increasing literacy levels by the use of linguistic prediction. *Child Language Teaching and Therapy* 8, 2 (June 1992), 138–187. https://doi.org/10.1177/026565909200800203

[30] Per Ola Kristensson. 2018. *Statistical Language Processing for Text Entry*. Vol. 1. Oxford University Press. https://doi.org/10.1093/oso/9780198799603.003.0003

[31] Antti Oulasvirta, Jussi P. P. Jokinen, and Andrew Howes. 2022. Computational Rationality as a Theory of Interaction. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 359, 14 pages. https://doi.org/10.1145/3491102.3517739

[32] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do People Type on Mobile Devices?: Observations from a Study with 37,000 Volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, Taipei Taiwan, 1–12. https://doi.org/10.1145/3338286.3340120

[33] Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM Press, New York, NY, USA, 83–88. https://doi.org/10.1145/2858036.2858305

[34] Dmitry Rudchenko, Tim Paek, and Eric Badger. 2011. Text Text Revolution: A Game That Improves Text Entry on Mobile Touchscreen Keyboards. In *Pervasive Computing*, Kent Lyons, Jeffrey Hightower, and Elaine M. Huang (Eds.).

Vol. 6696. Springer Berlin Heidelberg, Berlin, Heidelberg, 206–213. https://doi.org/10.1007/978-3-642-21726-5_13 Series Title: Lecture Notes in Computer Science.

[35] Andrew Swiffin, John Arnott, J. Adrian Pickering, and Alan Newell. 1987. Adaptive and predictive techniques in a communication prosthesis. *Augmentative and Alternative Communication* 3, 4 (Jan. 1987), 181–191. https://doi.org/10.1080/07434618712331274499

[36] Keith Trnka, John McCaw, Debra Yarrington, Kathleen F. McCoy, and Christopher Pennington. 2009. User Interaction with Word Prediction: The Effects of Prediction Quality. *ACM Trans. Access. Comput.* 1, 3, Article 17 (feb 2009), 34 pages. https://doi.org/10.1145/1497302.1497307

[37] Keith Trnka, Debra Yarrington, John McCaw, Kathleen F. McCoy, and Christopher Pennington. 2007. The Effects of Word Prediction on Communication Rate for AAC. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers (NAACL-Short '07)*. Association for Computational Linguistics, USA, 173–176. event-place: Rochester, New York.

[38] Keith Vertanen, Crystal Fletcher, Dylan Gaines, Jacob Gould, and Per Ola Kristensson. 2018. The Impact of Word, Multiple Word, and Sentence Input on Virtual Keyboard Decoding Performance. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–12. https://doi.org/10.1145/3173574.3174200

[39] Keith Vertanen and Per Ola Kristensson. 2011. A versatile dataset for text entry evaluations based on genuine mobile emails. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI '11*. ACM Press, Stockholm, Sweden, 295. https://doi.org/10.1145/2037373.2037418

[40] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. 2015. VelociTap: Investigating Fast Mobile Text Entry using Sentence-Based Decoding of Touchscreen Keyboard Input. (2015), 10.

[41] Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain Text Entry on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2307–2316. https://doi.org/10.1145/2556288.2557412

[42] Seunghak Yu, Nilesh Kulkarni, Haejun Lee, and Jihie Kim. 2018. On-Device Neural Language Model Based Word Prediction. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Santa Fe, New Mexico, 128–131. https://aclanthology.org/C18-2028

APPENDIX

A.1 Computation of speed metrics

We define different speed measures (all in keystrokes per second) to be able to disentangle cognitive and motoric processes in our performance analysis. Table 4 specifies the keystrokes whose inter-key intervals are included in the computation of each metric.

Measure	Keystrokes
Natural speed	• Must belong to words which were typed without the use of ITE's (i.e. no autocorrection or suggestion was used). • Must be a lowercase letter. • Must be preceded by a lowercase letter. • Must belong to words that are shorter than 5 letters. • Must belong to words that have a frequency higher than 100,000 [17].
Letter-only speed	• Must be a lowercase letter. • Must be preceded by a lowercase letter.
Letter-and-selection speed	• Must be a lowercase letter. • Must be preceded by a lowercase letter. OR • Is the keystroke representing the selection of a word from the suggestion list.
Overall speed	• All keystrokes are included.

Table 4. Description of keystrokes that are included in the computation of the different speed metrics.

A.2 Detection of suggestion usage strategies

Table 5 describes the criteria for detecting a strategy from the typing log.

A.3 Evaluation of the selection model

For estimating P_S, we use a logistic regression model. We chose logistic regression because it is probabilistic in nature. instead of making making a binary prediction whether a user would choose an offered suggestion or not, it outputs a continuous probability, which lends itself well to be used alongside the language model. To train the model, for each strategy, we perform a One vs. Rest analysis. *Fixup* and *space insertion* strategies were excluded since they occurred too infrequently. The features we include in the model are summarized in Table 6. They are based on our empirical findings and our domain knowledge. Some features were left out for certain strategies if they were not relevant. In particular, *leadup_speed* and *leadup_length* are not relevant for *prediction*, in which case no characters are typed yet for the currently suggested word. Similarly, for *contraction* and *capitalization*, we exclude *leadup_length*, since these strategies require the user to type the entire word before selecting the suggestion.

Strategy	Detection criteria	Comment
Completion	edit_distance is greater than 0 AND edit_distance equals len_diff AND The previous keystroke belongs to the same word as the current keystroke	A completion where even just one letter was corrected is classified as a correction. E.g. completing "reps" into "responsibility" is considered a correction due to the initial one-letter spelling mistake.
Correction	Case 1: edit_distance does not equal len_diff AND The previous keystroke belongs to the same word as the current keystroke. Case 2: Was classified as completion AND len_diff == 1 AND Last character of the previous text field is equal to the last character of the current text field AND The last two letters of the current text field are not equal	In the case where correction causes a len_diff equal to edit_distance (i.e. letters are added), we misclassify it as a completion. Case 2 partially corrects this by checking that a completion caused the last character of the text field to change. But this only corrects the cases where len_diff == 1. That being said, a correction causing an edit_distance and len_diff greater than 1 is rare.
Prediction	Only one keystroke in the word AND Last character of the previous text field is a space	
Contraction	edit_distance == 1 AND len_diff == 1 AND The selected word from the suggestion list contains an apostrophe AND The previous word string does not contain an apostrophe	
Capitalization	edit_distance == 0 AND len_diff == 0 AND The letters typed prior to the selection are all lower case AND The word selected from the suggestion list contains a capital	
Fixup	Preceded by a backspace AND The previous keystroke belongs to the same word as the current keystroke AND The last character of the previous text field is not a space	This does not currently catch words that were fully erased. Moreover, due to limitations in the web backend, we can only detect instances where the user made changes to the end of the word. Therefore, changes made after placing the cursor in the middle of the word are not detected.

Table 5. Rules for classifying strategies.

To determine the weights of these features, we trained the model on 90% of the dataset, excluding a random set of 10% of the 84,156 words for testing. In addition, we excluded words with only one character (e.g. *I*) and words with more than 14 characters which are too rare to be used for training. We also excluded words that were selected accidentally, i.e. where the selected suggestion differs from the final word that was typed.

A.3.1 Feature importance The coefficients of the logistic regression for each strategy are shown in Table 7. The significance and magnitude of the coefficients provide insights into the features and

their goodness for predicting whether a suggestion will be used. Thus, they allow us to gain further insights into which types of words each strategy is used and how user behavior differs between them.

The coefficients for word frequency, word length, and lead-up speed confirm our previous empirical findings that *completion* and *correction* are used on infrequent words. Only *prediction* is used for more frequent words. Similarly, the coefficients confirm that suggestions are used for longer words, where *word_length* in particular explains the use of *completion*. The *lead_up* speed indicates that suggestions are typed slower also before the suggestion is chosen, except for *contractions* which are typed somewhat faster before using a suggestion, likely because the user does not need to perform additional keyboard events to enter the apostrophe but can use the suggestion algorithm to insert it.

The *leadup_length* shows that users type fewer letters leading up to a *completion* than they do leading up to a *correction*. This confirms our earlier analysis of the edit distance associated with each strategy (Figure 5).

Some features we assumed to be relevant turned out to be less important. We still included them here to inform future work. We assumed, for example, touch offset might be positively correlated with using a suggestion since it represents touch keystrokes in areas of the screen that are inconvenient to reach. However, the influence of the offset on suggestion usage is very small in this model. Thus, discomfort might not motivate users to select suggested words or another metric might be more representative of discomfort during typing. Likewise, we find only a small effect of the number of backspaces. This might indicate that users are not motivated by the error-proneness of a word when using suggestions or they are unable to anticipate how error-prone a word is.

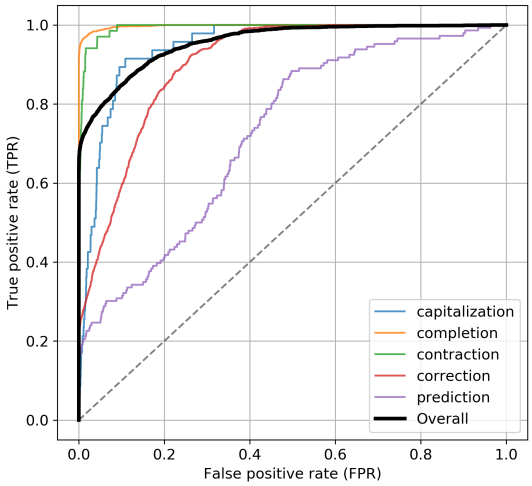


Fig. 8. Performance of the selection model.

A.3.2 Model performance for predicting suggestion use We excluded 10% of the data from training to use for testing the performance of our model. The logistic regression model outputs a probability that a word was typed by selecting a suggestion. This probability can be converted into a binary prediction by setting a threshold to distinguish between a manually typed word and a selected suggestion. Based on such binary predictions we evaluate the performance of the model. Note that the binary prediction is only used here to get an idea about the performance of the model. The

Feature name	Description	Relevance
word_length	The length, in number of characters, of the word.	Longer words may be more complex or take longer to type and thus suggestions are more likely to be used.
frequency	The log frequency of the word. This represents how common the word is in the English language [17].	Like word length, frequency is a proxy for the complexity of a word [9].
base_speed	Average characters per second across all instances of the word being typed manually (i.e. without ITEs). Includes only words typed more than 20 times.	Words that are complex or difficult to type would take a long time to type (on a character basis).
touch_offset	The average offset between the touchpoint and the center of the key. Data provided by Buschek et al. [7].	Words that involve pressing keys with larger offsets may be associated with more errors or more discomfort while typing.
backspaces	The average number of backspaces used when the word was typed normally in the dataset. Only words that were typed more than 20 times are included.	Words that on average incur more backspacing might indicate that the word is difficult to type or spell.
is_contraction	Whether the word is a contraction or not. Determined based on whether it contains an apostrophe.	Typing an apostrophe often requires additional keystrokes and therefore users may prefer to use suggestions to automatically insert apostrophes.
is_capitalized	Whether the word contains capital letters.	Capitalizing letters requires at least one additional keystroke (e.g. the "shift" key), and therefore users may prefer to use suggestions to automate capitalization.
leadup_speed	The typing speed, in characters per second, for the characters, typed so far.	A slower typing speed may reflect a user inspecting the suggestion list while typing, and therefore may indicate that the user intends on selecting a suggestion.
leadup_length	The number of characters typed before a suggestion is chosen.	Strategies vary in the number of letters being typed relative to the overall word length.

Table 6. List of word features used in the selection model.

application suggested in section 6 uses the continuous probability and does not define any fixed threshold.

Figure 8 visualizes the performance of our selection model as receiver-operating characteristic (ROC) curves. We use this ROC curves for a visual evaluation of the overall performance of the selection model and each strategy separately. These ROC curves show the performance of the model at all thresholds. Put briefly, the curves can be interpreted by visually estimating the area under the curve (AUC); the bigger the AUC, the better the performance of the model.

	Capitalization	Completion	Contraction	Correction	Prediction
const	-11.14*** (0.17)	-8.3*** (0.05)	-12.96*** (0.58)	-5.87*** (0.02)	-6.42*** (0.03)
word_length	0.37*** (0.04)	**5.58*** (0.03)**	-0.09 (0.1)	0.77*** (0.01)	0.53*** (0.04)
frequency	-0.61*** (0.07)	-0.41*** (0.02)	0.2 (0.11)	-0.39*** (0.01)	0.51*** (0.04)
is_contraction	-0.7 (1.13)	0.53*** (0.09)	**12.14*** (0.61)**	-1.05*** (0.1)	0.01 (0.9)
is_capitalized	**5.4*** (0.17)**	-0.24*** (0.07)	-1.44 (10.84)	-0.6*** (0.05)	-0.08 (0.25)
base_speed	0.53*** (0.05)	0.06*** (0.02)	-0.13** (0.06)	0.12*** (0.01)	0.14*** (0.03)
touch_offset	0.08*** (0.04)	0.04*** (0.01)	0.05 (0.06)	-0.04*** (0.01)	-0.11*** (0.02)
backspaces	-0.0 (0.03)	-0.25*** (0.02)	-0.2*** (0.03)	-0.7*** (0.01)	-0.57*** (0.05)
leadup_speed	**-1.76*** (0.09)**	**-3.37*** (0.04)**	0.26*** (0.06)	**-1.97*** (0.02)**	N/A
leadup_length	N/A	**-5.47*** (0.03)**	N/A	-0.3*** (0.01)	N/A

Table 7. Coefficients of the selection model for each strategy. *** = p < 0.01, ** = p < 0.05; non-sig. in gray, $|c|$ > 1 in bold.

Across strategies, the model performs better to predict *completion* and *contraction* with a rather large AUC, while the predictive power for the strategy *prediction* is rather limited showing a smaller AUC, corresponding to the low coefficients of the features for this strategy. Since distributions of strategy usages varied in the dataset some curves show steps, for example *contraction* and *capitalization*. With more data, these curves would become more smooth.

To provide a more specific example, we set the threshold to 0.05 to investigate the performance at a point that is close to the "elbow" of the ROC curve. At this threshold, the inherent probability that a user selected a suggestion is quite low. As described in Section 5.2.1, for example, a very long word only has a 25% chance of completion. At a threshold of 0.05, the accuracy of predicting a selection is 92.6%, and the precision and recall are 32.9% and 81.0%, respectively. With this threshold, our model is thus reasonably good at predicting the use of a suggestion with a low rate of false negatives, but with a high rate of false positives.

This evaluation shows that our model performs reasonably well and could be used in combination with existing suggestion algorithms. In our envisioned application, the purpose of the selection model is not to make a binary prediction but to estimate a continuous probability to rank words based on their likelihood of being suggested.

Received January 2023; revised May 2023; accepted June 2023

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die von mir angegebenen Quellen und Hilfsmittel verwendet habe.

Weiterhin erkläre ich, dass ich die Hilfe von gewerblichen Promotionsberatern bzw. –vermittlern oder ähnlichen Dienstleistern weder bisher in Anspruch genommen habe, noch künftig in Anspruch nehmen werde.

Zusätzlich erkläre ich hiermit, dass ich keinerlei frühere Promotionsversuche unternommen habe.

Bayreuth, den

Florian Lehmann