LanguageLogger: A Mobile Keyboard Application for Studying Language Use in Everyday Text Communication in the Wild

FLORIAN BEMMANN, LMU Munich, Munich, Germany
DANIEL BUSCHEK, Research Group HCI + AI, Department of Computer Science, University of Bayreuth, Bayreuth, Germany

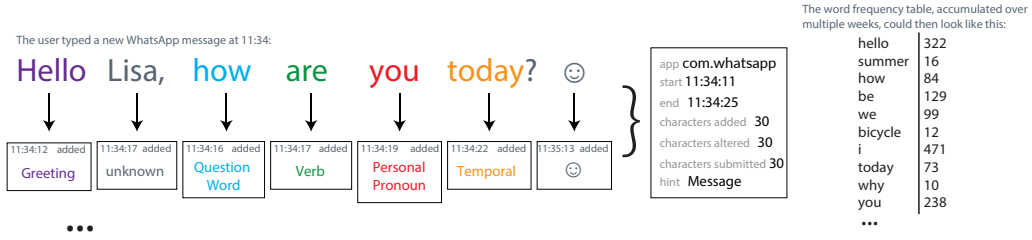

Fig. 1. Overview of our data logging concept to facilitate privacy-respectful studies of language use in everyday mobile text communication: Text entered by a participant (e.g. in a chat app) is abstracted to avoid revealing private content to the researchers while still catering to a wide range of common research interests. *Left:* Our *Word Categorisation* concept maps a predefined set of words to categories (e.g. "Hello"→"Greeting"). Moreover, *Custom Regex Filtering* allows for flexibly logging predefined strings, such as emojis. *Centre:* Metadata about the keyboard session is logged as well. *Right:* Our *Whitelist Counting* concept logs total usage counts for words in a predefined whitelist.

We present a concept and tool for studying language use in everyday mobile text communication (e.g. chats). Our approach for the first time enables researchers to collect comprehensive data on language use during unconstrained natural typing (i.e. no study tasks) without logging readable messages to preserve privacy. We achieve this with a combination of three customisable text abstraction methods that run directly on participants' phones. We report on our implementation as an Android keyboard app and two evaluations: First, we simulate text reconstruction attempts on a large text corpus to inform conditions for minimising privacy risks. Second, we assess people's experiences in a two-week field deployment (N=20). We release our app as an open source project to the community to facilitate research on open questions in HCI, Linguistics and Psychology. We conclude with concrete ideas for future studies in these areas.

CCS Concepts: • **Human-centered computing** → **Field studies**; **Text input**; **Smartphones**;

Keywords: Touch keyboard; data logging; language use; linguistics.

Authors' addresses: Florian Bemmann, LMU Munich, Munich, Germany, florian.bemmann@ifi.lmu.de; Daniel Buschek, Research Group HCI + AI, Department of Computer Science, University of Bayreuth, Bayreuth, Germany, daniel.buschek@uni-bayreuth.de.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2573-0142/2020/6-ART84 $15.00
https://doi.org/10.1145/3397872

ACM Reference Format:
Florian Bemmann and Daniel Buschek. 2020. LanguageLogger: A Mobile Keyboard Application for Studying Language Use in Everyday Text Communication in the Wild. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 4, EICS, Article 84 (June 2020). ACM, New York, NY. 24 pages. https://doi.org/10.1145/3397872

1 INTRODUCTION

Today, several lines of interdisciplinary research at the intersection of HCI, Psychology, and Linguistics aim to study people's language use in everyday computer-mediated communication (CMC), such as mobile chat apps. For example, researchers investigate the impact of communication means on language [3, 51] and language variations across ethnographies [21, 39, 44]. Other work analyses associations with personality of users of smartphone [47] and social networks [16, 24, 55].

Collecting data on language use is a key challenge central to all such work. Reseachers either examine existing corpora or they have to collect data from CMC applications on their own [51]. This comes with an inherent methodological challenge: There is a difficult tradeoff between a) limiting data collection to respect participants' privacy, and b) collecting comprehensive, unbiased and natural data to answer open research questions. Recent studies highlight this: For instance, Rosenfeld et al. [39] report that reluctance to participate was a key challenge for their study of communication via WhatsApp, despite emphasising encryption and anonymisation.

No ideal solution has emerged yet: Some studies aim for control in the lab and/or with writing tasks (e.g. "[w]rite an e-mail [...] explaining that you will not be able to take the next exam." [10]). This avoids observing private messages yet clearly limits the data to the given topics. To collect actual everyday messages, researchers have asked people for retrospective submissions of selected chat-logs. However, this data may be biased, for example due to capturing only one channel (e.g. one app, such as WhatsApp) and participants' criteria for selecting the submitted chats [15, 46, 49, 51].

It can also be difficult for researchers to avoid looking at private content during analysis of natural text logs (e.g. when thematically coding statements). This also extends to third-parties (e.g. chat partners in an app) who might not have consented to the study at all. As examples of private content, full chat logs may contain sensitive information like names, phone numbers, intimate conversations, passwords, and financial data.

We present a novel method and tool to address this challenge and support researchers in running studies of language use in the wild: LanguageLogger, a mobile keyboard app, allows researchers to log useful abstracted language data from text entered on participants' smartphones (Figure 1). We also provide this functionality as a logging module for integration into other research apps. Concretely, we employ three text analysis methods in a novel way, namely for *text abstraction that runs directly on participants' phones*. Hence, our app never reveals raw text to the researchers yet caters to many research interests. In brief, our text abstractions are:

- *Whitelist Counting:* The number of occurrences is logged for each word in a predefined whitelist, for example *"conference: 14", "paper: 25"*.
- *Word Categorisation:* Entered words are mapped to categories configured by the researchers. For instance, *"Jane is happy"* might be logged as *[name, verb, positive adjective]*.
- *Custom Regex Filtering:* Strings that match predefined regular expressions are logged as-is, or as an event indicating the occurrence of a match, but not the string itself. For instance, this can be used to log emoji use.

Note how these abstractions not only avoid recording potentially private text content but also pre-process the data in a way that many researchers require anyway. In summary, we contribute a novel concept and tool to enable studies of language use in everyday mobile communication, implemented as an Android keyboard app, with three text abstraction methods. To the best of

our knowledge there is no published tool to collect such data without recording full text or using artificial text tasks. With our work we thus aim to facilitate novel research in disciplines such as HCI, Linguistics, Communication Science, and Psychology. Our keyboard app and the logging module itself are available as an open source project.

2 RELATED WORK

We relate our work to 1) studies on language use, 2) methodology for assessing such language use in mobile contexts, and 3) approaches and tools from studies on text input.

2.1 Research Interests in Studying Language Use

Studying people's language use in CMC often involves multiple fields such as HCI, Psychology, and Linguistics. Our own work is motivated by such a long-term collaboration. Here we give an overview of several research questions to illustrate the importance and impact of innovating methods and tools for language data collection.

Fundamental research questions include the assessment of language use and the impact of technology [3], variations in language across different groups of people [21, 44], and related differences in communication behaviour [39]. Further research examines associations or inference based on language use, most prominently relating it to demographics and personality of users of mobile devices [47] and social networks [16, 24, 55].

The vision of such inference is an example where interests from multiple disciplines meet: Psychology is interested in predicting personality from everyday behaviour to extend or eventually even replace personality assessment based on self-reports [47]. HCI seeks to predict user personality to create more intelligent, personalised, and empathic systems [53], for example for chatbots [56]. With our keyboard app as a device-wide data logging platform we cater to both, comprehensive study data collection as well as future developments for integration into interactive intelligent systems.

Other current research asks for *long-term* assessments of language use, for example motivated by changes with age [42]. Another goal is to assess *all* means of communication, given that associations of language and other variables depend on context (cf. [12, 42]). This includes mobile communication (e.g. chat apps) targeted by our concept and tool.

Furthermore, current debates concern, for example, whether negative sentiment in written language really indicates negative mood [26]. Here it would be insightful to consider more than the currently analysed Facebook posts. This move from web sources to more personal chat communication appears as a recent general trend among research in this context and strongly emphasises the need for privacy-respectful methodology [39]. Our new logging approach facilitates such studies.

2.2 Existing Methodology to Assess Mobile Language Use

We give an overview of existing common approaches for (mobile) text data collection (see Table 1).

2.2.1 Collecting Text Created in Controlled Study Tasks. Participants are asked to compose text, such as writing an email to a fictitious person as part of a scenario, typically in the lab [10]. This avoids privacy issues since people do not need to reveal actual private text. However, the resulting data is severely limited in external validity, since behaviour and language use might not match that in real life. Repeated long-term observations are also cumbersome since people would need to return to the lab and/or these tasks several times.

2.2.2 Crawling Existing (Public) Text Content. As highlighted by Verheijen and Stoop [51], it is very difficult to obtain non-public language data, and thus researchers commonly rely on public sources such as blogs, forums, or posts on social networks [2, 48, 55]. Sometimes this data is enriched:

Data collection method	Description	Pros	Cons	Example Studies
Studies with text creation tasks	Participants compose text in a controlled study task and scenario, e.g. writing an email to a fictitious person.	no personal/private text involved	low external validity; limited to given scenario; repeated and long-term observation difficult / high effort; no language data from private communication	[10, 31]
Crawling existing (semi-)public content	Researchers collect language data from (semi-)public sources such as blogs, forums, or posts on social networks.	potentially large datasets; "natural" text not created for a study	limited to the specific sources available; no language data from private communication	[2, 16, 48, 55]
Collecting communication logs	Volunteers provide their chat logs, typically using existing export functionality e.g. of messenger apps.	language data also from personal mobile comm.; easy setup; potentially large datasets; "natural" text not created for a study; participants control what is seen by researchers	involves people not asked for consent (e.g. chat partners); biased by participants' selection / redaction; review & selection extra effort for participants	[11, 28, 49, 51]
Intercepting communication apps	Researchers intercept or listen to a chat (with consent), e.g. via an intermediate server or chatbot.	language data also from personal mobile comm.; "natural" text not created for a study	potential changes in behaviour by feeling "listened to"; technical efforts; need for recruiting pairs/groups of participants	[15, 46]
Wearable audio/video recorders; screen recordings	Participants wear a device that records audio and/or video, or keep a screen recorder running on their devices.	continuous logging of many comm. channels; potentially rich real-life context info; "natural" comm.	text not directly logged; includes people not asked for consent (e.g. bystanders); limited sample sizes; privacy requires participants' selection / redaction (i.e. effort and potential biases); manual data analysis required	[4, 29]

Table 1. Overview of common methods for collecting data on (mobile) text communication and language use, with key benefits and drawbacks. Note that the listed example studies are meant as examples for the *general* method as generalised for discussion here, i.e. not all pros/cons may apply to all of them, depending on other choices, constraints, etc. in the respective studies.

For instance, Yarkoni [55] asked bloggers to fill in a personality assessment questionnaire for later analysis in combination with language use on their public blogs. Golbeck et al. [16] and Kosinski et al. [24] applied related approaches for data collected via Twitter and Facebook, respectively.

These approaches are limited to those specific sources. Language use differs between personal messaging and shared online content and datasets from several sources show inherent differences [14]. The current trend towards studying personal communication such as chats particularly heightens privacy concerns [39]. Hence, we see our work in this paper as complementary, enabling privacy-respectful language data collection from new sources, namely 1) mobile devices in general, and 2) more personal contexts in particular, such as chat apps.

2.2.3 Asking People to Share Self-Selected Communication Logs. Personal (mobile) text communication can also be studied by asking volunteers to provide their chat logs to the researchers. Technically, this typically relies on export functionalities for end-users provided by many communication apps. For example, Ueberwasser et al. [49] created a multilingual corpus by asking WhatsApp users to send in their chat histories via email. Similarly, Verheijen et al. [51] asked participants to

provide WhatsApp chats in combination with access to their Facebook posts. Further variations include asking people to submit their last X messages [11] or all messages of a day [28]. To respect privacy, these studies require participants to first manually redact their messages, that is, to remove any private information they do not wish to reveal to the researchers.

This approach avoids artificial study tasks yet may have other drawbacks: First, the chat data typically includes content from chat partners who also need to consent to the study. Second, the collected chat data is often limited in its comprehensiveness and distorted by people's selection criteria: The researchers may get only a subset of the original data, without knowing how much or which data has been removed. Finally, review and selection may cause considerable extra effort for participants.

In contrast, our keyboard app inherently only logs text entered by the consenting device owner (assuming personal, unshared devices) and applies text abstraction directly on the device to respect privacy without manual reviewing or selection.

2.2.4 Intercepting Communication Apps and Chat Rooms. Another way to collect CMC data in the wild is to listen to a chat with consent. For example, participants may agree to send messages not directly to their partners but first to the researchers' system [46], which then forwards them. Researcher can also add themselves or a bot to the conversation, who just listens passively [15]. Song et al. [46] combined both in their live chat collection.

This avoids artificial tasks yet risks biasing collected data in that people may be likely influenced by awareness of the listeners (e.g. avoiding more private topics of conversation). An added bot or researcher might also change one-to-one communication to a group chat which in itself might change user experience, the UI (e.g. group chat view), and thus user behaviour.

2.2.5 Beyond Written Communication. Our focus is on written communication, yet related methods also can be found in research on verbal communication: Mehl et al. [29] provided people with a wearable device that recorded 30 second sections of audio every 12 minutes. The data was analysed by manual transcription. Respecting privacy might be challenging when researchers have to listen to potentially sensitive recordings. Mehl et al. addressed this by allowing participants to review and select their recordings first. This has also been used in studies using screen recordings of smartphone use [4]. The potential downsides again include extra effort for participants and possibly introducing selection biases. In contrast, our approach of text abstraction eliminates the need for this. Future work could enable our approach for audio/video logging by using a speech-to-text system to transcribe the text before applying our text abstractions, all directly on the device.

2.3 Studies of Text Input Behaviour

We propose a keyboard app for logging language use. HCI researchers have studied how people interact with mobile keyboards to enable fast and accurate typing [25]. Studying *how* users enter text is different from studying *what* people type, that is, language use. However, some challenges are common to both questions, in particular the need for suitable methods to go beyond lab studies [20, 38]: Most importantly, many text entry studies used transcription tasks, in which participants copy given text. This avoids privacy concerns since no private text is observed [38]. However, such studies cannot assess "natural" behaviour because content, time, and situation of text input is set by the researchers [6].

To overcome this and improve external validity, Buschek et al. [6] recently presented a concept and keyboard app that enables researchers to collect users' typing touches throughout natural daily smartphone interactions. Privacy concerns were addressed with a subsampling filter approach, which only recorded text-revealing data for a small subset of keypresses. In this way, no readable private messages were recorded.

For studying language use we cannot use this method [6], since it was built on the idea of *avoiding* to log whole words. Hence, we propose a novel concept that reconciles privacy-respectful logging with the data requirements arising from research questions about everyday language use. Next, we describe our concept in detail.

3 CONCEPT DEVELOPMENT PROCESS

Overall, we employ a keyboard app with three text abstraction methods. Here we describe our concept development process.

3.1 Defining Foundations for Supporting Privacy & Trust

At the outset, our need for a new data logging concept was motivated by our practical experiences with several interdisciplinary studies on data collection in the wild. We found that trust in such field study setups is more complex than a yes/no decision with typical informed consent and approvals. Thus, taking additional measures to make a study setup more privacy-respectful in our view is a highly worthy investment. This also follows the foundations presented in related work on privacy-aware keyboard logging [6]: The goal is to support a trustful relationship between participants and researchers with a concept and tool that records relevant data without revealing private content.

Note that, as in the work by Buschek et al. [6], this assumes a generally trustworthy foundation: We do not claim full protection against malicious intent (e.g. researchers actively trying to spy on their participants) since ethical research standards still need to be upheld by the researchers and in more fundamental, institutional ways – and not exclusively by a keyboard app.

3.2 Assessing Logging Requirements for Research

To support a wide range of research interests we analysed the literature in detail. Our goal was not to conduct an exhaustive survey but to identify the main study approaches, as presented in the related work section. We analysed the work with regard to requirements for both logging procedures and the resulting logged data to answer these questions: 1) Which data is observed (e.g. chat messages)? 2) Which measures are computed on said data (e.g. word counts)? 3) Which privacy measures/challenges are reported (e.g. participants reviewing messages)?

In addition, we ran an interdisciplinary workshop on logging language use and a series of subsequent discussions with four researchers from HCI, Psychology and Statistics. This was organised in the context of a joint long-term research project and study planning, thus eliciting real methodological needs. We also discussed our literature analysis with this group.

We aggregated the following requirements for our mobile tool:

- *Integration in everyday life:* To observe natural everyday language without bias the logging tool must be incorporated in people's usual environment/systems.
- *Differentiation between shared-public and private content:* Language differs between personal messaging and publicly shared content. Thus, our tool should allow for assessing or filtering for the context of writing (e.g. in which app).
- *No extra efforts for participants:* To enable long-term deployment in the wild with reasonable effort, the system should not demand extra work from participants, such as reviewing logs.
- *Device-wide logging – but none for chat partners:* To yield a comprehensive corpus and respect privacy, the tool should be able to process all mobile writing of the participant, but none of other people (e.g. chat partners).
- *No access to raw text:* To respect privacy, no one except the participant should be able to access the raw typed text.

Text Analysis	Study Goal	Data	Covered by
Word-based: no. of words/textisms (e.g. "lol")	Use of textisms in formal vs informal comm. [10]	Email written in study task	Whitelist Counting
Category-based: no. of words per pre-defined category (LIWC cat. [32])	Associations of word use and personality [55]	Public blog articles	Word Categorisation
Category-based (LIWC [32]); word-based: psycho-linguistic stats on counts of words in a 150k dictionary (MRC [54])	Associations of tweets and personality [16]	Public twitter data	Word Categorisation, Whitelist Counting
Message-based: no. of words/characters; keyboard-based: use of auto-correction & suggestion	Use of WhatsApp / chat behaviour [49]	WhatsApp chat histories	Keyboard Sessions, Further Logging
Word-based: no. of words/textisms; interaction-based: texting speed/frequency	Relating texting speed/frequency with language & literacy measures [31]	Mobile texting (in given study task)	Whitelist Counting, Keyboard Sessions, Further Logging
Non-textual cues: counting emojis	Associations of emoji use and personality [27]	Public twitter data	Custom Regex Filtering

Table 2. Examples of common text analysis methods in research on (mobile) language use, along with example studies and data sources. The last column indicates which of our text abstractions and logging features cater to each analysis. Overall, the table illustrates that we address a wide range of research interests: We enable these analyses for data from everyday (personal) mobile text communication, while avoiding that people have to share actual raw text with the researchers. Note that some measures were self reported (e.g. use of auto-correction [49], texting frequency [31]), whereas we enable quantitative logging of such data.

3.3 Developing a Logging Strategy: Text Abstraction

From our requirements analysis it became evident that in the vast majority of studies the actual raw text content is ultimately rarely needed for the conducted text analysis. For example, text is often analysed in abstracted and aggregated ways, such as the number of words per category. Table 2 lists key examples to illustrate this. Therefore, a viable approach to facilitating both participants' privacy and the researchers' workflow is to apply these text data processing steps directly on the device, to then only share the resulting abstracted data. A closer look motivates three particular types of text abstractions:

Words are counted: For instance, the number of occurrence of words or word categories is analysed with regard to associations with the author's personality [16, 55] or to compare language use between contexts (e.g. formal vs informal [10]).

Words are categorised, that is, mapped to pre-defined categories or values: For example, Herring et al. [21] was interested in 'female' and 'male' stylistic words. There also exist generalised categorisations that are widely used across studies, for example the LIWC dictionary [32]. "Categories" can also be numeric, such as SentiWS by Remus et al. [37], which assigns sentiment scores to words.

Specific terms are measured: Some studies analysed aspects of text that require more flexibility. Typically, this involves text elements not captured by categorisations such as LIWC [32]. However, these are still analysed in an abstracted way, namely as occurrences or counts. A prominent example are recent analyses of emoji use [27, 52].

4 FINAL LOGGING CONCEPT

We next describe our final concept in detail.

4.1 Text Abstractions

First, we describe how we realised the three kinds of text abstraction motivated above. For a visual overview, see Figure 1.

4.1.1 Whitelist Counting. For this text abstraction method, the researchers define a list of words before the study. Whenever one of these whitelisted words is entered this is counted by the system.

For example, consider a study that looks at the use of positive words such as *good, happy, great* and thus includes these in the whitelist. The sentence "Thomas is happy" would then be logged as "happy: 1". Other words, such as the name "Thomas", are not recorded.

These counts are summed up, such that at the end of a study, the researchers in this example might get a table such as "good: 123; happy: 456; great: 789". It is easy to see that this does not reveal private messages as long as enough text is logged overall. We quantify this with our experiments.

4.1.2 Word Categorisation. Here, researchers define a mapping from words to categories (e.g. "happy: positive emotion"). Words are then mapped to categories and the system only logs that those categories occurred (we call these *word events*). If a word is not included in the mapping dictionary, the category *unknown* is recorded.

As metadata, category log entries also have a timestamp, the app, a flag indicating if the word was added/edited/removed, and a reference to a keyboard session. In case of an edit, the category before and after the change is logged.

Overall, entered text is thus logged as a sequence of such word events. For example, when a user types "I am happy" in a WhatsApp message this might be logged as three word events: *personal pronoun, verb, positive emotion*. This does not reveal the raw text as long as categories are not too specific (i.e. in an extreme case where each word is its own category this would log raw text). Again, we quantify this in our experiments.

4.1.3 Custom Regex Filtering. We integrate a regular-expression matcher to allow for logging custom patterns that would be hard to specify in a list of terms, as for the other two abstraction methods. Entered text is checked against the configured matcher(s). In case of a match, the system can either 1) log the matched string as-is or 2) just the occurrence of a match.

For example, this could be used to count how often people enter phonenumbers (without logging them), or to log emojis via the emoji unicode range: "Call me at 0123456789 😁" would then be logged as two *regex events*, that is, "phonenumber" and "😁", plus metadata (e.g. timestamp, app).

4.2 Lemmatisation

Before applying the text abstraction concepts described above, each word can optionally be lemmatised. For instance, *went, going* would be replaced with the root word *go*. This is useful since common dictionaries such as LIWC [32] do not contain inflected forms.

4.3 Keyboard Sessions

We define a keyboard session to start when the keyboard opens and end when it is closed. We store metadata for each session: Number of characters added/altered, name of the app in which the text has been entered, hint-text/label of the text field, timestamps of the session's start and end.

4.4 Further Logging

Further logged data includes the use of word suggestions and auto-corrections (occurrences, not words). The app can log available Android sensor data which was also logged by related work [6], such as touch data (not text revealing), device orientation, accelerometer, gyroscope, and so on.

4.5 Logging Exceptions

Following related work [6], we realised two logging exceptions: First, we never log fields for passwords, logins, addresses etc. This uses Android field types. While we cannot guarantee that every field is marked in this way by developers, many apps do so since it is essential for accessibility. Moreover, note that strings like passwords and names of accounts/people are not part of our whitelist dictionary and thus never logged as text anyway. Second, as the related work, our keyboard UI shows a small lockpad icon that allows people to pause logging.

5 IMPLEMENTATION AS AN ANDROID MODULE AND KEYBOARD APP

We created a reusable and remotely configurable Android keyboard application, LanguageLogger, as described next.

5.1 Overview of System Modules

For reusability, the LanguageLogger app project consists of four modules: One is the keyboard, the other three realise the data processing. Each module represents one layer of functionality. This loose coupling allows easy integration of the implemented functionalities (or a subset thereof) into other applications, such that our logging methods can be used with other studies and apps, not only as part of our keyboard. The embedding into Android applications ensures that raw text content does not leave the client device. The dependencies between the four modules are visualized in Figure 2.

Regarding the "flow" of the data, the application module provides the raw source data and passes it on to one or both of the processing modules. The resulting data then is returned to the application module. Thus no data is exchanged between the other LanguageLogger modules implicitly.

In the following four subsections we present the four modules in more detail, each representing one layer of functionality.

5.2 Application Module (here: Keyboard)

The Application Module constitutes the "host app" that uses LanguageLogger logic to process its logdata. Here we build on the keyboard app of Buschek et al. [6], which in turn uses Google's Android Open Source Project (AOSP) Keyboard[1].

The host app module includes the other required modules as local library module dependencies[2]. A module's functionality can then be used by instantiating the respective class and calling their methods. The result is either provided as return value (synchronous operation) or can be obtained by implementing a callback and passing it with the method call for asynchronous operations. For flexibility, the storage or transmission of the resulting log data also has to be implemented in this host app module; in the case of our keyboard app implementation this includes the transmission to the LanguageLogger server. We visualize the dataflow between the modules in Figure 3.

[1] https://android.googlesource.com/platform/packages/inputmethods/LatinIME/, *last accessed 12th May 2020.*
[2] https://developer.android.com/studio/build/dependencies#dependency-types, *last accessed 12th May 2020.*

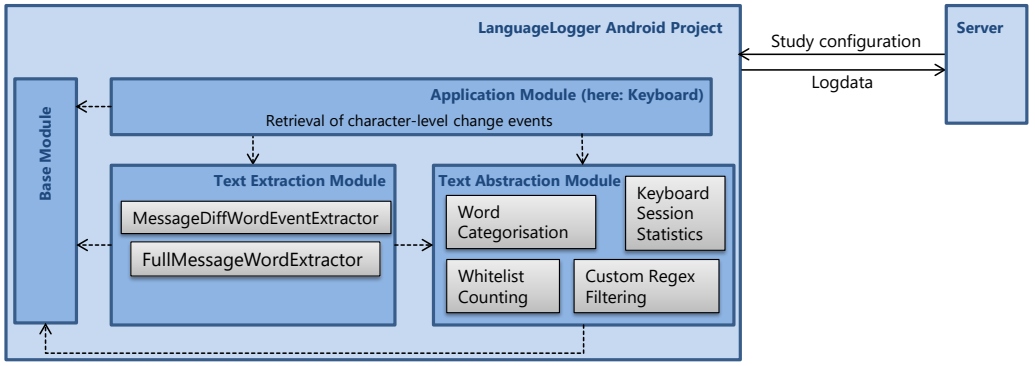

Fig. 2. Overview of the main architecture of our app: Dashed arrows indicated Android module dependence. The preprocessing (*Text Extraction Module*) and abstraction logic (*Text Abstraction Module*) are separated from the host application (*Application Module*). All modules are loosely coupled, thus it is possible to use the LanguageLogger logic in other Android applications as well.

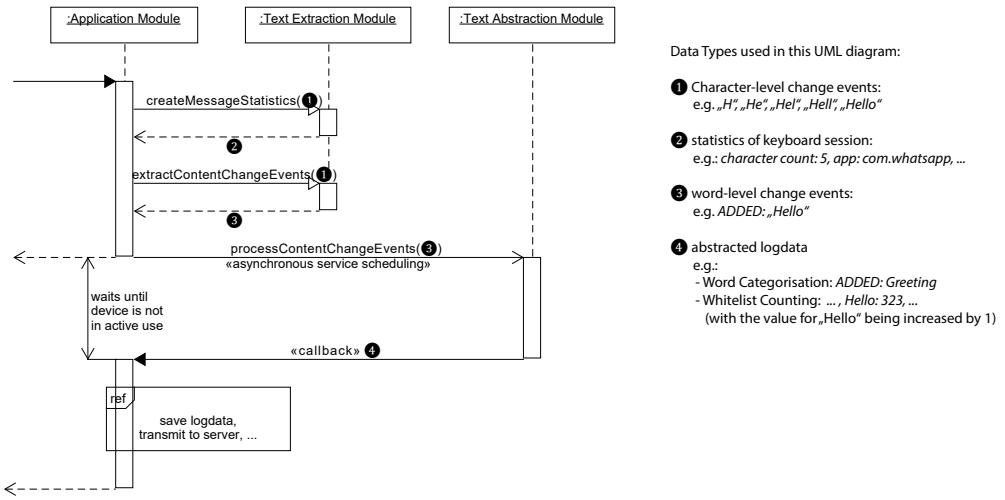

Fig. 3. An UML sequence diagram visualizing the dataflow between the LanguageLogger modules. Which types of data are passed is indicated by the numbers in the filled circles, which are explained on the right. Typing the word "Hello" is used as exemplary user action.

5.3 Base Module

Functionality that is required by multiple other modules is located in the base module, to avoid redundancy. This includes a REST client that matches our LanguageLogger server implementation, and utility classes for handling console logs.

5.4 Text Extraction Module

The structure of the input data collected by applications depends on the data source. Thus, this module servers as a layer to separate text abstraction logic from source-specific preprocessing steps.

5.4.1 ContentChangeEvents. Language log data typically arises in the form of character-level logs (e.g. when observing user interactions). In the case of our keyboard, for example, events consist of the text field content after each keystroke: For instance, a user typing the word "Tom" in a sentence "Hello Tom" yields the events *"Hello T"*, *"Hello To"*, *"Hello Tom"*.

However, to apply our text abstractions, we need events on word-level, such as in this case *ADDED: "Tom"* . In the following, we refer to such an event as a *ContentChangeEvent*.

In particular, we distinguish two types of such events, fitting the two types of Word Categorisation described in Section 4.1.2:

- *ADDED, CHANGED, REMOVED, SPLIT, JOINED*, represent user actions (on words) that happened in temporal order during a typing session.
- *CONTAINS* corresponds to the presence of a word in the final submitted text (e.g. a text message entered in a chat app).

These *ContentChangeEvents* are created from incoming character-level events using the class *MessageDiffWordEventExtractor* (for events of type *ADDED, CHANGED, REMOVED, SPLIT, JOINED*) and *FullMessageWordExtractor* (*CONTAINS*). This works synchronously, as the processing is not computationally expensive.

5.4.2 FullMessageWordExtractor. This can be implemented in a straightforward fashion: It takes the text and splits it into single words with a regular expression. In particular, this expression matches sequences of word characters (e.g. in our implementation for German: a-zA-Z0-9äöüß'), enclosed by non-word characters (everything except the word characters).

5.4.3 MessageDiffWordEventExtractor. The extraction of *ADDED, CHANGED, REMOVED, SPLIT* and *JOINED* events is more complex. The general algorithm, in short, works as follows. For each character change event:

- Split the text into words, with the regular expression mentioned above.
- Identify the word that has changed or was added/removed: Compare each word in the content after the character change with the word at the same index before the character change.

All consecutive character change events affecting the same word are collected. Comparing the content before the first and after the last character change event of such a sequence yields the overall change that the user performed.

However, there are many edge cases of user actions that cannot be tackled without significantly extending this general procedure. To name just a few:

- Adding a new word in between existing ones: We observed users technically doing this by appending or prepending a word to an existing word and typing the separating space character only at the very end. For example, the sequence *"Hello how are you"*, *"Hello Thow are you"*, *"Hello Tohow are you"*, *"Hello Tomhow are you"*, *"Hello Tom how are you"* would, without special consideration, result in *CHANGED: "how"* → *"Tomhow"*, *SPLIT: "Tomhow"* → *"Tom, how"*. However, the intended correct log should just be: *ADDED: "Tom"*
- Pasting and auto-correction: If users paste text sequences or auto-correction changes complete words multiple characters (possibly at different positions) may change with one change event published by the system, whereas the general algorithm described above expects just one character to differ.

We conducted early user tests, including raw content alongside the extracted words, to identify as many of these edge cases as possible. Thereafter, we used test-driven development to improve our algorithm, adding unit tests for each newly identified edge case. In this way, we extended our first general algorithm to a carefully grown decision tree.

5.5 Text Abstraction Module

This module implements the language processing of Section 4 in a resource-efficient manner that does not disturb the user. In contrast to the work performed in the Text Extraction Module, the Text Abstraction Modules is more computationally intensive: The word lists and category mappings defined by the researchers are loaded into working memory and are repeatedly queried. Thus, the required memory and computation time depends on the study configuration and can become quite large (e.g. the German dictionary we deployed in our study contains 300,000 words). We implemented the following measures to still keep the processing unobtrusive for the user:

- Load one list after another: The jobs are not parallelized and word lists are not combined to one single list. This lowers the peak memory requirement.
- Implementation as Android AsyncTask[3]: The work is performed on a background thread, avoiding blocking the calling thread.
- Resource-aware scheduling with Android JobScheduler[4]: The AsyncTask is wrapped by a JobService, that is scheduled to launch when the device is not in active use.

If lemmatisation is activated it is applied in this module. In our implementation, we used the TreeTagger software of Schmidt et al. [41], but this can be flexibly changed. All services for the text abstraction methods can be scheduled with one method call, through the class *RIMEInputContentProcessingController*. Due to the underlying asynchronous implementation, callbacks have to be implemented to obtain the resulting data.

Furthermore, the Text Abstraction Module contains a service class to calculate aggregated statistics of keyboard sessions (*KeyboardMessageStatisticsGenerator*). For example, this includes calculating the number of characters entered in the keyboard session and the keyboard session start/end time. Other applications could use this logic to get similar statistics for their respective unit of reference (e.g. for notification logging this would compute statistics per notification).

5.6 Backend

We implemented a backend application communicating with the mobile app through a REST interface. The backend fulfills two main objectives: Configuration of the text abstractions, and storage of the logdata.

5.6.1 Text Abstraction Configuration. The text abstractions presented in Section 4 can be configured in a system of phyical and logical configurations: Researchers upload a word-to-category mapping (for Word Categorisation) and a wordlist (for Whitelist Counting), as a physical mapping/list. One or multiple physical lists can then be combined into a single logical configuration. These logical configurations are then used on the mobile device. This separation allows for easy changeability, for example, in the case that a single dictionary needs to be replaced or updated within the context of a larger study configuration. Moreover, Custom Regex Filtering is configured by entering regular expressions in the backend. The LanguageLogger app downloads these configurations from the backend during the setup process.

[3] https://developer.android.com/reference/android/os/AsyncTask, *accessed 6th May 2020.*
[4] https://developer.android.com/reference/android/app/job/JobScheduler, *accessed 6th May 2020.*

5.6.2 Storage of Log Data. The log data that the backend retrieves from the mobile devices is stored in a relational database, implementing the relations between keyboard sessions and extracted categories. Each category belongs to a keyboard session. Custom Regex Filtering logs are also treated as categories, as their data structure is similar. The whitelist word counts are collected in an absolute frequency table, encompassing the whole study duration.

6 USER STUDY

We ran a field study as an example deployment of our text abstraction methods and tool. Our aim was to test the methods and app and to assess the users' views of the text abstractions. An additional technical evaluation of key parameters of the abstraction methods is available in Appendix A.

6.1 Apparatus: Keyboard App, Questionnaires, Web UI

For the study we set our keyboard to this example configuration: For Word Categorisation, we used the common LIWC dictionary [32]. For Whitelist Counting, we used the DeReWo Lemma List published by the Leibniz Institute of the German Language (IDS) [1]. For Custom Regex Filtering, we specified matchers for emojis. These are example choices for this first deployment, motivated by the literature. Our tool is flexible and easy to use with other dictionaries, whitelists, and so on. Apart from our keyboard, we used the following components:

A *web interface* showed participants examples of their own logged data (e.g. word categories, counts). It was used as part of the post-study questionnaire and the interviews.

A *pre-study questionnaire* explained the three text abstraction concepts in detail, along with illustrated examples (similar to Figure 1). This not only served as an introduction but also as part of the informed consent procedure. The questionnaire also assessed people's attitudes towards privacy in general, using the item sets of Buchanan et al. [5]. Finally it asked for demographics and provided instructions to install our Android app.

A *post-study questionnaire* asked about 1) potential influences of the keyboard and logging on usability and experience, 2) perceived privacy protection, and 3) feedback on app and study. Moreover, the questionnaire linked to the web UI and thereafter again asked about perceived privacy protection.

6.2 Participants

We recruited 20 participants (12 female, 7 male, 1 prefer not to disclose) via newsletters and social media. Their mean age was 24.5 years (range 19 - 34). They received €5, plus €5 for an optional interview (which 4 people did). Using the items of Buchanan et al. [5], people's self-reported attitude regarding privacy concern, general caution, and the importance of technical protection was slightly above neutral.

6.3 Procedure

Participants first filled in the pre-study questionnaire. Then they installed our app on their phones and set it as their default keyboard. After two weeks of use, they filled in the post-study questionnaire, including the web interface that showed examples of their logged data. Finally, we invited participants to semi-structured interviews to get a more detailed picture on the points from the questionnaires. For example, we asked about their impressions of the study and app, perceived privacy protection with the text abstractions, and further feedback. Interviews were audio recorded to facilitate later analysis.

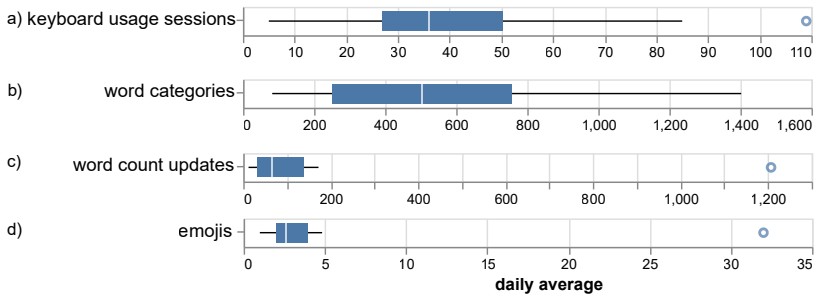

Fig. 4. Overview of our study data, averaged per day and person: (a) Number of logged keyboard sessions, (b) word events, (c) increases of word counts (Whitelist Counting), and (d) regex events (here: emojis). Each boxplot shows the median, upper- and lower quartile, min/max whiskers and outliers.

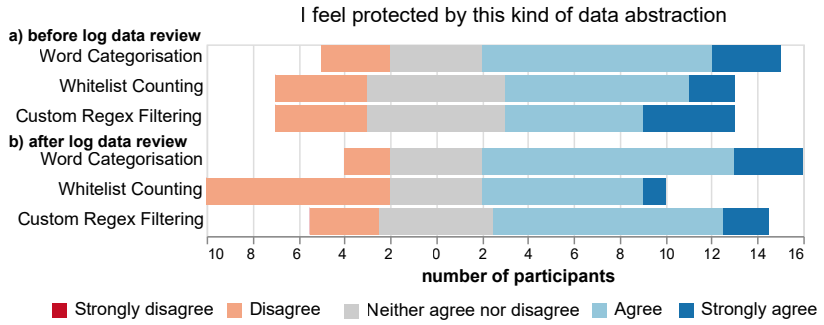

Fig. 5. Results from the Likert questions on the feeling of privacy protection by the three logging concepts, (a) before the log data review and (b) after it.

6.4 Results

For this report, we define a *word event* as entering a word, editing a word, or deleting a word. We define a *keyboard session* as the time from opening the keyboard to closing it.

6.4.1 Overview of Logged Data. We logged a total of 12,318 keyboard sessions. On average, a session had 10.16 (SD 6.39) word events. Averaged per day and person, we logged 42.86 (SD 26.69) keyboard sessions, 539.56 (SD 345.61) categorised words by Word Categorisation, 172.65 (SD 266.01) count updates by Whitelist Counting, and 4.42 (SD 6.98) emojis by Custom Regex Filtering (also see Figure 4). Regarding typing context, the top five apps were: WhatsApp (6,455 keyboard sessions), Chrome browser (940), Instagram (435), Google Quicksearch on the homescreen (404), and Telegram messenger (376).

6.4.2 Perceived Privacy through Text Abstractions (Post-Study Q.). Our post-study questionnaire included the five-point Likert item *"I feel protected by this kind of data abstraction"* (Figure 5 top, 5=strongly agree). Here, participants rated Word Categorisation best (M=4), followed by Custom Regex Filtering (M=3.5) and Whitelist Counting (M=3.5).

We also asked people to order the text abstractions by how much these contribute to respecting their privacy: 13 of 20 people judged Word Categorisation to contribute the most to respecting privacy, followed by Whitelist Counting (4) and Custom Regex Filtering (3).

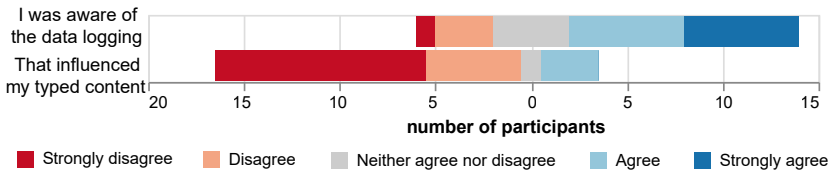

Fig. 6. Results on awareness of logging while typing and the reported influence of that awareness on the typed content.

6.4.3 Influence of Log Data Review (Post-Study Q.). The post-study questionnaire presented users with a subset of their log data via a web UI, then asked again about privacy protection through the abstractions (Figure 5 bottom). We found no significant changes comparing ratings before and after this review for the three text abstractions (Wilcoxon signed-rank tests, all $p > .05$). Hence, we found no evidence of an overall shift towards lower or higher ratings.

There were rather individual shifts, both ways: 16 of 20 people changed their opinion on at least one of the three concepts. Ratings for Whitelist Counting were changed by seven people, who all except for one lowered their ratings. In the interviews, two of those stated that they could relate the counting data more easily to themselves than expected. One participant noticed a high count for "I", making him wonder whether he was too egocentric. Another one recognised terms from band names and concluded that he might be identified by these otherwise rare words if his musical taste was known. For Word Categorisation we did not notice a trend (8 did not change, 6 increased, 6 decreased). For Custom Regex Filtering five people changed their rating (4 increases, 3 decreases).

6.4.4 Influence of Study Setting on Text Content (Post-Study Q.). Our post-study questionnaire had five-point Likert items on potential influences of the study situation and the keyboard functionality and UI. Figure 6 shows the results: Overall, people reported that they were aware of the data logging during the study (median 4). However, they indicated that this had little to no influence on the text they entered (median 1).

6.4.5 Influence of Study Keyboard on Usability (Post-Study Q.). Our post-study questionnaire also included five-point Likert items on potential differences of our app to people's usual keyboard apps. Most people found differences (median M=4) and felt that this influenced their interaction behaviour (M=4). More specifically, 16 people (rather) agreed that different auto-completion was an influence (M=5), while ten rated this way for visual UI differences (M=3). Free text questions and interviews showed that auto-completion was worse since the app had not learned users usual words (yet). Moreover, a key UI difference was how to access special characters and emojis.

7 LIMITATIONS

Our text abstraction methods and tool enable new studies of everyday mobile language use in the wild which have not been possible so far. Nevertheless, our approach comes with some limitations:

Our app only logs text entered by the participant. Hence, we can neither easily study full conversations, nor distinguish between receivers of text within one app. Extensions might, for example, extract chat partners from notifications (cf. [40]), although this raises issues of logging data from non-participants.

It is also difficult to use text abstractions that require word/category lists in open vocabulary work [43], which explores occurring terms instead of a defined set. This may also apply to slang terms. This could be partly addressed by defining a very comprehensive whitelist, or by informing word lists with a pre-study (e.g. to collect slang terms).

Furthermore, text abstractions inherently limit what kind of text representations are available. For instance, many computational methods in Natural Language Processing use word embeddings (i.e. words represented as high-dimensional vectors, e.g. [33]): In a privacy view, these embeddings are no different to logging words, since usually embeddings can be turned back into words. Hence, we do not support directly logging words as embeddings. However, the word data that is indeed logged with our tool (e.g. whitelisted words) could be converted to embeddings post-hoc (albeit not contextual ones [45]). Regarding further word metrics, a future version might be extended, for example, to count word co-occurrences for the whitelisted words.

Finally, introducing a keyboard app is unlikely to match people's own keyboards exactly. We used a common open-source keyboard from Google to optimise familiarity for many Android users. Nevertheless, people noticed differences for auto-correction and small UI design choices. While we do not expect these to considerably impact on language use, we plan to improve on this in future work (e.g. via options for UI customisation and importing existing user dictionaries).

8 DISCUSSION

8.1 Logging Everyday Mobile Language Use

We have shown that it is feasible to log everyday mobile language use in a privacy-respectful way, using a keyboard app with integrated text abstractions. Researchers can further minimise the potential of reconstructing raw text from such abstracted data by logging enough data per person. In our experiments and study, 50-100 sentences comes close to the minimum and can easily be achieved within a two-week study (for many people in our study much earlier, 1-2 days; also see Appendix A).

Data collected with our method and tool can meet many sought-after criteria, such as 1) covering all mobile text communications, 2) including personal ones (e.g. chats), 3) unobtrusive long-term measurements, and 4) varied natural everyday contexts. Very few people felt that the study influenced the content of their writing. People in our study also typed in their usual everyday apps, including chat apps and web browsers. These results are all positive with regard to achieving comprehensive and unbiased data collection. We thus conclude that our method and tool present a valuable addition to the toolset for research intersted in everyday mobile language use.

8.2 Remaining Privacy Risk

As in related work [6], our goal is to facilitate privacy-respectful studies – not to counter malicious attacks. Responsible study planning still comprises more than a keyboard (e.g. secure data storage). Besides attacks, it might be possible to gauge a user's interest in some topics from word counts (e.g. one interviewee mentioned rare words over-represented in his favourite band names). If this is to be avoided it can be addressed with a more selective whitelist for word counting. Overall, the choice of whitelist and category mappings influences what can be inferred from the abstracted data. Our app supports tailoring these settings to research questions, which can limit collected data and possibly unwanted inference opportunities. In all cases, our text abstractions avoid logging raw text.

8.3 Privacy Perception

The majority of participants found that the text abstraction methods contributed to protecting their privacy in our study. In more detail, perception of privacy was individual: In the interviews, some participants stated quite low concern and did not deem it necessary to use our provided opportunity for a log data review. Others were more sceptical and would have preferred more details about the logging system's inner workings. Fittingly, participants' scores also varied for

the questionnaire on general privacy concerns [5]. Individuality of perspectives is also indicated by the fact that the log data review influenced participants' perception of the abstractions in both directions (see next discussion point).

8.4 Log Data Review

In contrast to other logging methods, ours does not require participants to manually review data as text abstractions are applied automatically. We still tested the idea of data review, using a web UI after the study. Seeing their own abstracted data influenced some people's perceptions of the abstraction methods yet not in a systematic way overall. One exception was Whitelist Counting, which four people found less protective after the review. Crucially, motivations for rating protection more critically after seeing the data were not related to fears of raw text being reconstructed, indicating that our method succeeded with regard to this main concern. Instead, people noticed potential inferences of a more general kind (e.g. one interviewee wondered if a high count of "I" indicated egocentrism). Feedback generally showed that people liked this view on their data. Overall, we conclude that showing actual logged data could replace generic examples to inform participants of how the data is processed. Future work could also integrate such a view into the keyboard app directly.

8.5 Communicating Privacy-Aware Logging

Two people avoided entering passwords and locations. We do not record password fields and neither passwords nor specific location names are in the whitelist. Fittingly, one interviewee explained that she might not have behaved differently if she had better understood how the system worked exactly. Here, we see a tradeoff between 1) providing extensive (technical) detail to facilitate trust, and 2) not overwhelming people (cf. long terms of use). Future work could integrate some explanations into the app as part of a typical first-launch intro, which users know from many apps today.

8.6 Further Opportunities for Research and Applications

We illustrate the rich opportunities enabled by our text abstractions and tool with ideas for future studies, methods, and applications.

8.6.1 Analysing the Use of Non-Verbal Cues in Mobile Messaging. An active line of research analyses non-verbal cues, such as emojis, for example to reveal misunderstandings [8, 30] and improve UIs [35, 36]. Recent work using questionnaires [52] concluded that real-world data collection is needed yet warned about privacy challenges due to logging private messages. Our tool enables such studies: For instance, researchers can set regular expressions that capture emojis in context (cf. [35]).

8.6.2 Long-term Observations of Everyday Language Use. Running silently in the background our logging module enables long-term observations without repeatedly asking people to do study tasks. For example, such studies could investigate changes in language use when a person moves to a new city, country, or social circle, and/or starts to learn and use a new language. Related work also indicates changes with age [42].

8.6.3 Enriching Data Collected with Other Methods. Our approach can be combined with other mobile study methods, such as experience sampling (ESM) or mobile (context) sensing: Studies could use ESM to ask users about subjective experiences (e.g. mood) and now relate this to in-situ language use collected with our tool at and around that moment. As an example, we have already integrated the logging module into another app, which also includes ESM, in a project at the

intersection of HCI and Psychology. More generally, such combinations of different objective and subjective information channels enrich observations [7, 50].

8.6.4 Informing Intelligent Text Entry Systems. Word suggestion and correction algorithms could use datasets on language use collected with our logging module, for example, to address the cold-start problem for new users of a keyboard. Moreover, intelligent reply systems (e.g. Google Smart Reply [22]) could consider a user's word usage to generate replies that are in line with personal style and language use.

8.6.5 Personalising Intelligent Mobile Applications. Our logging module could also be integrated into further applications: For instance, a chatbot might prefer words that fit the user's logged frequent vocabulary to avoid misunderstandings. Related, educational apps could use our logging module to help users learn a foreign language, for example by estimating progress from words used in that language or by teaching translations for the user's used words, increasing relevance for the learner (cf. [34]).

8.6.6 Enhancing Logging Beyond the Keyboard. Our logging module could also be integrated into other logging applications that deal with textual data different from keyboard logs, such as mobile notifications. This could enhance previous studies which looked only at notification categories [40], e.g. to now also analyse content topics and sentiment.

8.6.7 Integration with Other Privacy-Facilitating Approaches. There also exist other approaches to collect data in the wild in an anonymous way: For example, differential privacy [9] and randomized response techniques [13] add systematic noise to the data, without changing the full dataset's characteristics. However, such noise might be less suitable for studying language use of individual users, which is important for research in Psychology. Such different approaches could also be combined: For example, differential privacy for a specific use case could be applied on top of our tool, e.g. as noise on our extracted word frequencies.

9 CONCLUSION

Several lines of research are interested in people's language use in everyday mobile typing, including work in HCI, Linguistics and Psychology. Unfortunately, studies on language use in the wild suffer from a tension between a) limiting data collection to facilitate participants' privacy and b) collecting comprehensive, unbiased and natural data to answer open research questions.

As the main contribution of this paper, we presented and evaluated a keyboard app which integrates three novel customisable text abstraction methods that run directly on participants' phones. A key conceptual insight here is that such abstractions facilitate privacy by directly pre-processing the data in a way that many researchers require anyway for their analyses. For the first time, this enables researchers to collect comprehensive data on language use during natural mobile typing without logging participants' private messages.

Data collected with our method and tool is attractive since it offers high external validity: First, it covers all means of text communication on the participant's device yet avoids logging data of non-participants. Second, this data can be collected unobtrusively in long-term deployments, capturing the full range of contexts that mobile device users encounter in their everyday lives. The literature shows that such data is important to address many open research questions at the intersection of HCI, Psychology, Linguistics, and Communication Studies.

We release our Android app and related material to the research community to facilitate future work on these topics:

https://www.medien.ifi.lmu.de/language-logger/

ACKNOWLEDGMENTS

This project is funded by the Bavarian State Ministry of Science and the Arts in the framework of the Centre Digitisation.Bavaria (ZD.B).

REFERENCES

[1] 2013. Korpusbasierte Wortgrundformenliste DEREWO, v-ww-bll-320000g-2012-12-31-1.0, mit Benutzerdokumentation. http://www.ids-mannheim.de/derewo

[2] Azy Barak and Orit Gluck-Ofri. 2007. Degree and Reciprocity of Self-Disclosure in Online Forums. *CyberPsychology & Behavior* 10, 3 (2007), 407–417. https://doi.org/10.1089/cpb.2006.9938

[3] Michael Beißwenger and Angelika Storrer. 2008. 21. Corpora of Computer-Mediated Communication. *Corpus Linguistics. An International Handbook. Series: Handbücher zur Sprach-und Kommunikationswissenschaft/Handbooks of Linguistics and Communication Science. Mouton de Gruyter, Berlin* (2008).

[4] Barry Brown, Moira McGregor, and Donald McMillan. 2014. 100 Days of iPhone Use: Understanding the Details of Mobile Device Use. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services* (Toronto, ON, Canada) *(MobileHCI '14)*. ACM, New York, NY, USA, 223–232. https://doi.org/10.1145/2628363.2628377

[5] Tom Buchanan, Carina Paine, Adam N. Joinson, and Ulf-Dietrich Reips. 2007. Development of Measures of Online Privacy Concern and Protection for Use on the Internet. *J. Am. Soc. Inf. Sci. Technol.* 58, 2 (Jan. 2007), 157–165. https://doi.org/10.1002/asi.v58:2

[6] 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* (Montreal QC, Canada) *(CHI '18)*. ACM, New York, NY, USA, Article 255, 14 pages. https://doi.org/10.1145/3173574.3173829

[7] Daniel Buschek, Sarah Völkel, Clemens Stachl, Lukas Mecke, Sarah Prange, and Ken Pfeuffer. 2018. Experience Sampling As Information Transmission: Perspective and Implications. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers* (Singapore, Singapore) *(UbiComp '18)*. ACM, New York, NY, USA, 606–611. https://doi.org/10.1145/3267305.3267543

[8] Henriette Cramer, Paloma de Juan, and Joel Tetreault. 2016. Sender-intended Functions of Emojis in US Messaging. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Florence, Italy) *(MobileHCI '16)*. ACM, New York, NY, USA, 504–509. https://doi.org/10.1145/2935334.2935370

[9] Irit Dinur and Kobbi Nissim. 2003. Revealing Information While Preserving Privacy. In *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (San Diego, California) *(PODS '03)*. Association for Computing Machinery, New York, NY, USA, 202–210. https://doi.org/10.1145/773153.773173

[10] Michelle Drouin and Claire Davis. 2009. R u txting? Is the Use of Text Speak Hurting Your Literacy? *Journal of Literacy Research* 41, 1 (2009), 46–67. https://doi.org/10.1080/10862960802695131

[11] Michelle Drouin and Brent Driver. 2014. Texting, textese and literacy abilities: A naturalistic study. *Journal of Research in Reading* 37, 3 (2014), 250–267. https://doi.org/10.1111/j.1467-9817.2012.01532.x

[12] Penelope Eckert. 2008. Variation and the indexical field1. *Journal of Sociolinguistics* 12, 4 (2008), 453–476. https://doi.org/10.1111/j.1467-9841.2008.00374.x

[13] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (Scottsdale, Arizona, USA) *(CCS '14)*. Association for Computing Machinery, New York, NY, USA, 1054–1067. https://doi.org/10.1145/2660267.2660348

[14] Golnoosh Farnadi, Geetha Sitaraman, Shanu Sushmita, Fabio Celli, Michal Kosinski, David Stillwell, Sergio Davalos, Marie-Francine Moens, and Martine De Cock. 2016. Computational personality recognition in social media. *User Modeling and User-Adapted Interaction* 26, 2 (01 Jun 2016), 109–142. https://doi.org/10.1007/s11257-016-9171-0

[15] Chris Fullwood, Lisa J. Orchard, and Sarah A. Floyd. 2013. Emoticon convergence in Internet chat rooms. *Social Semiotics* 23, 5 (2013), 648–662. https://doi.org/10.1080/10350330.2012.739000 arXiv:https://doi.org/10.1080/10350330.2012.739000

[16] J. Golbeck, C. Robles, M. Edmondson, and K. Turner. 2011. Predicting Personality from Twitter. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*. 149–156. https://doi.org/10.1109/PASSAT/SocialCom.2011.33

[17] Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages.. In *LREC*, Vol. 29. 31–43.

[18] 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)*. ACM, New York, NY, USA, 194–195. https://doi.org/10.1145/502716.502753

[19] Joshua T. Goodman. 2001. A bit of progress in language modeling. *Computer Speech & Language* 15, 4 (2001), 403 – 434. https://doi.org/10.1006/csla.2001.0174

[20] Niels Henze, Enrico Rukzio, and Susanne Boll. 2012. Observational and Experimental Investigation of Typing Behaviour Using Virtual Keyboards for Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. ACM, New York, NY, USA, 2659–2668. https://doi.org/10.1145/2207676.2208658

[21] Susan C. Herring and John C. Paolillo. 2006. Gender and genre variation in weblogs. *Journal of Sociolinguistics* 10, 4 (2006), 439–459. https://doi.org/10.1111/j.1467-9841.2006.00287.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9841.2006.00287.x

[22] 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)*. ACM, New York, NY, USA, 955–964. https://doi.org/10.1145/2939672.2939801

[23] R. Kneser and H. Ney. 1995. Improved backing-off for M-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1. 181–184 vol.1. https://doi.org/10.1109/ICASSP.1995.479394

[24] Michal Kosinski, David Stillwell, and Thore Graepel. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences* 110, 15 (2013), 5802–5805. https://doi.org/10.1073/pnas.1218772110

[25] 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)*. ACM, New York, NY, USA, 335–338. https://doi.org/10.1145/2628363.2628405

[26] Ethan Kross, Philippe Verduyn, Margaret Boyer, Brittany Drake, Izzy Gainsburg, Brian Vickers, Oscar Ybarra, and John Jonides. 2019. Does counting emotion words on online social networks provide a window into people's subjective experience of emotion? A case study on Facebook. *Emotion (Washington, D.C.)* 19, 1 (February 2019), 97—107. https://doi.org/10.1037/emo0000416

[27] Weijian Li, Yuxiao Chen, Tianran Hu, and Jiebo Luo. 2018. Mining the Relationship between Emoji Usage Patterns and Personality. In *International AAAI Conference on Web and Social Media*. AAAI Publications, Palo Alto, CA, USA, 4. http://arxiv.org/abs/1804.05143

[28] Rich Ling and Naomi S. Baron. 2007. Text Messaging and IM. *Journal of Language and Social Psychology* 26, 3 (2007), 291 –298. https://doi.org/10.1177/0261927X06303480

[29] Matthias R. Mehl, James W. Pennebaker, D. Michael Crow, James Dabbs, and John H. Price. 2001. The Electronically Activated Recorder (EAR): A device for sampling naturalistic daily activities and conversations. *Behavior Research Methods, Instruments, & Computers* 33, 4 (01 Nov 2001), 517–523. https://doi.org/10.3758/BF03195410

[30] Hannah Jean Miller, Daniel Kluver, Jacob Thebault-Spieker, Loren G Terveen, and Brent J Hecht. 2017. Understanding Emoji Ambiguity in Context: The Role of Text in Emoji-Related Miscommunication.. In *International AAAI Conference on Web and Social Media*. AAAI Publications, Palo Alto, CA, USA, 152–161.

[31] Gene Ouellette and Melissa Michaud. 2016. Generation text: Relations among undergraduates' use of text messaging, textese, and language and literacy skills. *Canadian Journal of Behavioural Science / Revue canadienne des sciences du comportement* 48, 3 (2016), 217–221. https://doi.org/10.1037/cbs0000046

[32] James W Pennebaker, Martha E. Francis, and Roger J. Booth. 2001. Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates* 71, 2001 (2001), 2001.

[33] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[34] Paul R. Pintrich. 2003. A Motivational Science Perspective on the Role of Student Motivation in Learning and Teaching Contexts. *Journal of Educational Psychology* 95, 4 (2003), 667–686. https://doi.org/10.1037/0022-0663.95.4.667

[35] Henning Pohl, Christian Domin, and Michael Rohs. 2017. Beyond Just Text: Semantic Emoji Similarity Modeling to Support Expressive Communication. *ACM Trans. Comput.-Hum. Interact.* 24, 1, Article 6 (March 2017), 42 pages. https://doi.org/10.1145/3039685

[36] Henning Pohl, Dennis Stanke, and Michael Rohs. 2016. EmojiZoom: Emoji Entry via Large Overview Maps. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Florence, Italy) *(MobileHCI '16)*. ACM, New York, NY, USA, 510–517. https://doi.org/10.1145/2935334.2935382

[37] Robert Remus, Uwe Quasthoff, and Gerhard Heyer. 2010. SentiWS-A Publicly Available German-language Resource for Sentiment Analysis.. In *LREC*. Citeseer.

[38] Shyam Reyal, Shumin Zhai, and Per Ola Kristensson. 2015. Performance and User Experience of Touchscreen and Gesture Keyboards in a Lab Setting and in the Wild. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. ACM, New York, NY, USA, 679–688.

https://doi.org/10.1145/2702123.2702597

[39] Avi Rosenfeld, Sigal Sina, David Sarne, Or Avidov, and Sarit Kraus. 2018. A Study of WhatsApp Usage Patterns and Prediction Models without Message Content. *CoRR* abs/1802.03393 (2018). arXiv:1802.03393 http://arxiv.org/abs/1802.03393

[40] Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber, and Albrecht Schmidt. 2014. Large-scale Assessment of Mobile Notifications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. ACM, New York, NY, USA, 3055–3064. https://doi.org/10.1145/2556288.2557189

[41] H. Schmid. 1999. *Improvements in Part-of-Speech Tagging with an Application to German*. Springer Netherlands, Dordrecht, 13–25. https://doi.org/10.1007/978-94-017-2390-9_2

[42] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin E P Seligman, and Lyle H Ungar. 2013. Personality, gender, and age in the language of social media: the open-vocabulary approach. *PloS one* 8, 9 (2013), e73791. https://doi.org/10.1371/journal.pone.0073791

[43] H. Andrew Schwartz, Johannes C. Eichstaedt, Margaret L. Kern, Lukasz Dziurzynski, Stephanie M. Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin E. P. Seligman, and Lyle H. Ungar. 2013. Personality, Gender, and Age in the Language of Social Media: The Open-Vocabulary Approach. *PLOS ONE* 8, 9 (09 2013), 1–16. https://doi.org/10.1371/journal.pone.0073791

[44] Beat Siebenhaar. 2006. Code choice and code-switching in Swiss-German Internet Relay Chat rooms. *Journal of Sociolinguistics* 10, 4 (2006), 481–506. https://doi.org/10.1111/j.1467-9841.2006.00289.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9841.2006.00289.x

[45] Noah A. Smith. 2019. Contextual Word Representations: A Contextual Introduction. *arXiv:1902.06006 [cs]* (Feb. 2019). http://arxiv.org/abs/1902.06006 arXiv: 1902.06006.

[46] Zhiyi Song, Stephanie Strassel, Haejoong Lee, Kevin Walker, Jonathan Wright, Jennifer Garland, Dana Fore, Brian Gainor, Preston Cabe, Thomas Thomas, Brendan Callahan, and Ann Sawyer. 2014. Collecting Natural SMS and Chat Conversations in Multiple Languages: The BOLT Phase 2 Corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. European Languages Resources Association (ELRA), Reykjavik, Iceland, 1699–1704. http://www.lrec-conf.org/proceedings/lrec2014/pdf/1094_Paper.pdf

[47] Clemens Stachl, Sven Hilbert, Jiew-Quay Au, Daniel Buschek, Alexander De Luca, Bernd Bischl, Heinrich Hussmann, and Markus Bühner. 2017. Personality Traits Predict Smartphone Usage. *European Journal of Personality* 31, 6 (2017), 701–722. https://doi.org/10.1002/per.2113

[48] Yla Tausczik, Kate Faasse, James W. Pennebaker, and Keith J. Petrie. 2012. Public Anxiety and Information Seeking Following the H1N1 Outbreak: Blogs, Newspaper Articles, and Wikipedia Visits. *Health Communication* 27, 2 (2012), 179–185. https://doi.org/10.1080/10410236.2011.571759

[49] Simone Ueberwasser and Elisabeth Stark. 2017. What's up, Switzerland? A corpus-based research project in a multilingual country. *Linguistik Online* 84, 5 (Sep. 2017). https://doi.org/10.13092/lo.84.3849

[50] Niels van Berkel, Denzil Ferreira, and Vassilis Kostakos. 2017. The Experience Sampling Method on Mobile Devices. *ACM Comput. Surv.* 50, 6, Article 93 (Dec. 2017), 40 pages. https://doi.org/10.1145/3123988

[51] Lieke Verheijen and Wessel Stoop. 2016. Collecting Facebook Posts and WhatsApp Chats. In *Text, Speech, and Dialogue*, Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala (Eds.). Springer International Publishing, Cham, 249–258.

[52] Sarah Theres Völkel, Daniel Buschek, Jelena Pranjic, and Heinrich Hussmann. 2019. Understanding Emoji Interpretation through User Personality and Message Context. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipeh, Taiwan) *(MobileHCI '19)*. ACM, New York, NY, USA. https://doi.org/10.1145/3338286.3340114

[53] Sarah Theres Völkel, Ramona Schödel, Daniel Buschek, Clemens Stachl, Quay Au, Bernd Bischl, Markus Bühner, and Heinrich Hussmann. 2019. *Opportunities and Challenges of Utilizing Personality Traits for Personalization in HCI: Towards a shared perspective from HCI and Psychology*. De Gruyter, Oldenbourg, Germany.

[54] Michael Wilson. 1988. MRC psycholinguistic database: Machine-usable dictionary, version 2.00. *Behavior Research Methods, Instruments, & Computers* 20, 1 (01 Jan 1988), 6–10. https://doi.org/10.3758/BF03202594

[55] Tal Yarkoni. 2010. Personality in 100,000 Words: A large-scale analysis of personality and word use among bloggers. *Journal of Research in Personality* 44, 3 (jun 2010), 363–373. https://doi.org/10.1016/j.jrp.2010.04.001 arXiv:NIHMS150003

[56] Michelle X. Zhou, Gloria Mark, Jingyi Li, and Huahai Yang. 2019. Trusting Virtual Agents: The Effect of Personality. *ACM Trans. Interact. Intell. Syst.* 9, 2-3, Article 10 (March 2019), 36 pages. https://doi.org/10.1145/3232077

A ADDITIONAL TECHNICAL SIMULATIONS

Here we report on additional technical experiments which simulate attempts at reconstructing raw text from text abstracted with our concepts. Note that these insights are not to be misunderstood as an "attack study", nor a "privacy guarantee". Our goal rather is to examine which conditions for study and logging parameters should be fulfilled, in particular with regard to giving a guideline for a minimum study duration / number of logged sentences before analysis.

A.1 Assumptions

We make the following assumptions: First, the reconstruction has access to the full LanguageLogger database, containing both logging data and configuration files, including the Word Categorisation mapping. Second, the first word of each sentence that should be reconstructed is already known.[5]

In particular, we examine these study and logging parameters:

- $N \in \mathbb{N}$, the total number of logged sentences.
- $c \in \{true, false\}$, indicating whether Word Categorisation is used or not. If it is not used, the logged data only consists of the total counts of whitelisted words from Whitelist Counting. If it is used, the logged data also contains the word events (see Final Concept Section and Figure 1).
- $n_{cat} \in \mathbb{N}$, the number of different categories configured for Word Categorisation.
- $r_{unk} \in [0, 1]$, the ratio of distinct words appearing in the raw text data for which no category is defined.

A.2 Dataset

Our reconstruction simulation could be run on any large text corpus. In particular, here we use 205,082 German sentences from the "mixed-typical German" corpus of Goldhahn et al. [17], which consists of text crawled from the news, web, and emails. We selected sentences that do not contain words which occur in less than 0.01% of all sentences in that dataset. This was motivated by reducing the number of distinct words overall to reduce computational costs. We chose a German corpus, since we had access to a German version of a widely used category mapping (LIWC [32]). We release our scripts to facilitate replication on other corpora.

A.3 Text Abstraction Settings

We use the set of all distinct words in the corpus as a whitelist (i.e. all words are counted), which serves as an upper bound of information on raw text that could be gained from word counts. For Word Categorisation, we use the widely used LIWC mapping [32], which has $\approx$ 400 categories (respectively 2438 unique category combinations in our setup). We do not specify any Custom Regex Filtering. Some of the following experiments use different settings, as stated where applicable.

A.4 Language Model

We employ a language model as part of the reconstruction. A plethora of options exist, from basic statistics to deep learning. As an exemplar for our experiments, we decided for a middle ground; an n-gram model (with n=7 and Kneser-Ney-Smoothing [19, 23]). This is also motivated by wide-spread use in related work (e.g. for predictive keyboards [18]). Intuitively, it takes (up to 6) previous words to then predict for a given next word how likely it is to follow thereafter. We trained the model on the corpus described above.

[5]Note that these assumptions are highly unrealistic to be fulfilled for a malicious attack – they imply that the attacker has access to the database and has somehow observed the beginning of the desired text (e.g. via shoulder-surfing) as an entry point for reconstruction. We make these assumptions to be able to even attempt reconstructions, not as an attack study.

A.5 Reconstruction Procedure

An experiment with (N, c, n_{cat}, r_{unk}) runs as follows:

A.5.1 Simulating Data Logging. A set of N sentences is randomly chosen from the corpus. We treat this data as if it was logged during a study, that is, we apply the text abstraction concepts to it.

A.5.2 Reconstructing Sentences. We then pick a random sentence from this "logged" data and attempt to reconstruct it. The first word is given as a starting point, then we repeat the following steps word by word:

(1) *Filter by Whitelist Counting:* Words with count greater than zero are considered as candidates to be the next word.
(2) *Filter by Word Categorisation:* Words are excluded whose categories do no match those logged for the next word.
(3) *Scoring:* Remaining words are scored by multiplying relative logged frequency with language model likelihood.
(4) *Decision:* The word with the highest score is predicted as the next word.
(5) *Count update:* The predicted word's count in the count table is reduced by one (i.e. drawing without replacement).

This procedure ends once the same number of words as in the original sentence have been generated (i.e. we simulate a reconstruction model with perfect stopping).

A.5.3 Repetition and Performance Measures. To reduce impact of randomness, we repeat this 1000 times. To quantify reconstruction success, we measure the percentage of correctly reconstructed sentences, and the edit distance between the correct and the reconstructed sentence on a per-word level.

A.6 Results

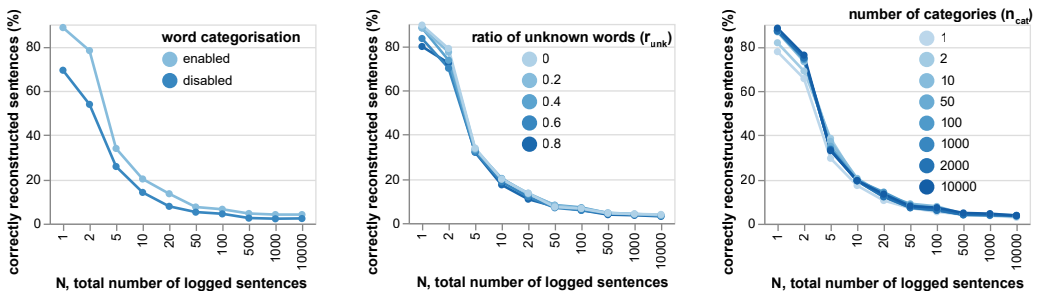

Fig. 7. Results of the reconstruction simulation, showing percentage of correctly reconstructed sentences depending on three parameters. *Left:* Word Categorisation on/off (c). *Centre:* Percentage of words unknown to the category mapping (r_{unk}). *Right:* Number of categories (n_{cat}).

Experiments took nine days of computing time on a Supermicro SuperServer SYS-1029GQ (28 cores, 768 GB RAM).

A.6.1 Influence of Number of Logged Sentences (N, c). First we compared different values of N and c (see Figure 7 left): For very low N (1 or 2) our model reconstructed most sentences correctly (88.6% with categories, 69.3% without). Increasing N leads to a steep drop in success. With Word Categorisation enabled, ($c = true$) success drops below 10% at $N = 50$; without, this happens already

at $N = 20$. For large values of N the percentage of correctly guessed sentences settles around 4% with categories and 2% without.

A.6.2 Influence of Category Mapping (n_{cat}, r_{unk}). We also compare the parameters of the word-category mapping. To do so, we created mappings with varying numbers of categories (n_{cat}) and varying ratios of unknown words (r_{unk}) by assigning ratios of r_{unk} words at random to n_{cat} artificial categories (e.g. "C1", "C2" etc.). For each such created mapping we then measured the reconstruction success.

Figure 7 (centre) plots the results: Higher r_{unk} result in lower reconstruction success, yet this effect is small: For $N = 1$ and $N = 2$, comparing low and high values of r_{unk} (i.e. compare the lines in the figure), we see differences in success of up to 10%. For $N = 5$ or higher these differences drop below 5%.

Increasing the number of categories n_{cat} slightly improves the percentage of correctly guessed sentences, as is to be expected (Figure 7, right): For low N, comparing values of n_{cat}, we see differences up to 10%. These differences between values of n_{cat} become nearly zero at $N = 500$.

A.7 Conclusion and Takeaway for Studies with our Method

Our results show that for very little data (very small N) the information from Whitelist Counting and Word Categorisation can restrict the space of potentially typed words.

The configuration of the Word Categorisation has only a minor impact compared to this. None of the tested setups considerably facilitated reconstruction. For example, the commonly used LIWC dictionary thus is clearly a suitable choice (its characteristics approximately correspond to $n_{cat} = 2000$ and $r_{unk} = 0.4$).

Note that the remaining absolute reconstruction success in these simulations is due to our strong assumptions, e.g. knowing the first word and the length of each sentence. This does not occur in practice (e.g. researchers using our tool have never access to the first word of each sentence). Moreover, for intuition, inspecting the word-level edit distances plus example cases showed that incorrect reconstructions are not "just slightly wrong" but indeed so far off that they do not relate to the original's content in any meaningful way.

Overall, when using our text abstractions and tool we recommend to conduct studies in a way such that at least 50 sentences of log data per participant can be expected.

Received February 2020; revised March 2020; accepted April 2020