SkipWriter: LLM-Powered Abbreviated Writing on Tablets

Zheer Xu[1], Shanqing Cai[2], Mukund Varma T[3], Subhashini Venugopalan[2], Shumin Zhai[2]
Dartmouth College[1], Google[2], UC San Diego[3]

UIST PITTSBURGH | 2024

Motivation

Handwriting is well-suited for **abbreviated phrase input.**

- Prolonged stress on the hand and wrist
- Flexible to modify/extend any abbreviation

Large Language Models (LLMs) have shown significant potential in decoding **ambiguous or partial inputs**.

Abbreviation Form

when would you come home

Each word in the phrase is abbreviated as a variable-length prefix.

Decoder Implementation

Approach: Recognize-then-decode (avoid costly data collection)

- **Recognizer**: In-production handwriting recognizer used in Gboard
- **LLM**: A fine-tuned checkpoint of PALM 2

mod l h (raw handwriting) → Recognizer → "mod l h" (user abbreviation) → LLM

"Large Language Models (LLMs) may offer transformative opportunities for text input, especially for physically demanding" (preceding context) → LLM

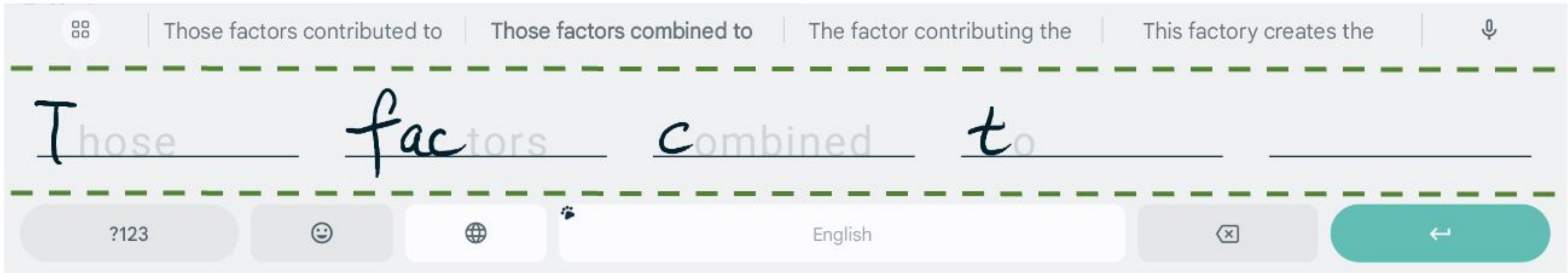
LLM → "modalities like handwriting" (intended phrase)

Data Synthesis for LLM fine-tuning

- Arbitrarily generate abbreviations for millions of phrases.
- Words with common prefixes are likely to end with longer abbrevs.
- Definition of ***Prefix Entropy*** and ***progressive generation*** approach:

$$H_{\text{prefix}} = -\sum_{w \in W} p(w)\log(p(w)) \qquad P_i = \frac{H(c_1, c_2, \ldots, c_i)}{H_0}$$

Interface Design



Similar to regular keyboard layouts

- **Top:** Candidate bar **Bottom**: Function keys
- **Middle**: Customized handwriting area

Segmented Rule: Automatic accommodation of word abbreviation

- Easy future completion
  - Reminds users to leave additional space for future completion.
  - Reduces the overhead of editing and encourages the user to try shorter abbreviations for maximal character savings.
- Low-cost word delimiters for robust prefix recognition
- Inline visualization of the top candidate to reduce attention switch

User Evaluation

Baseline: Handwriting keyboard on Gboard (non-abbreviated style).

Participants: Ten right-handed volunteers (9 male, 1 female)

Task: Transcribe 15 given sentences for each technique.

Test Set: Randomly sampled from the test split of 4 public datasets used for fine-tuning the decoder.

Apparatus: Participants use stylus to write on an Android tablet (Lenovo P12 Pro), with the LLM decoder remotely deployed.

	SkipWriter	Baseline
Speed	25.78 WPM	24.18 WPM
Word Error Rate	2.08%	4.05%
Traversal Distance Per Character	11.41 mm* (**60% ↓**)	18.74 mm*

Traversal Distance Per Character: the cumulative stylus traversal distance over the course of a test sentence divided by the count of non-whitespace characters in the committed full text

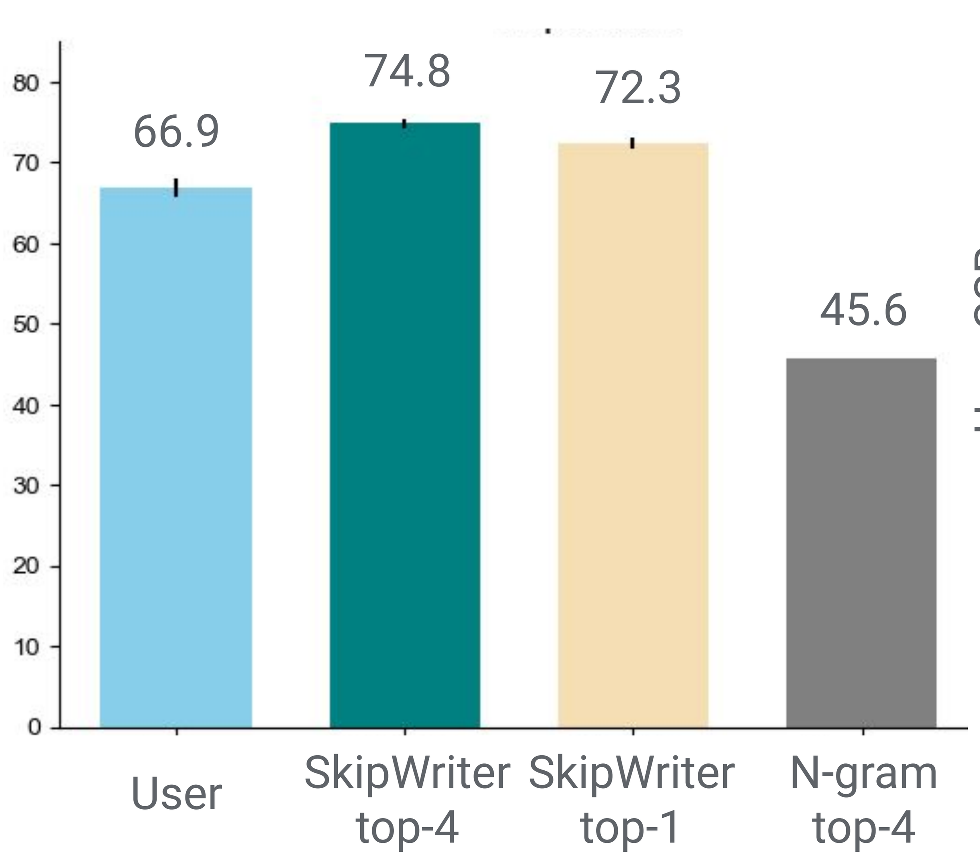
*: statistical significance observed

Offline Simulation

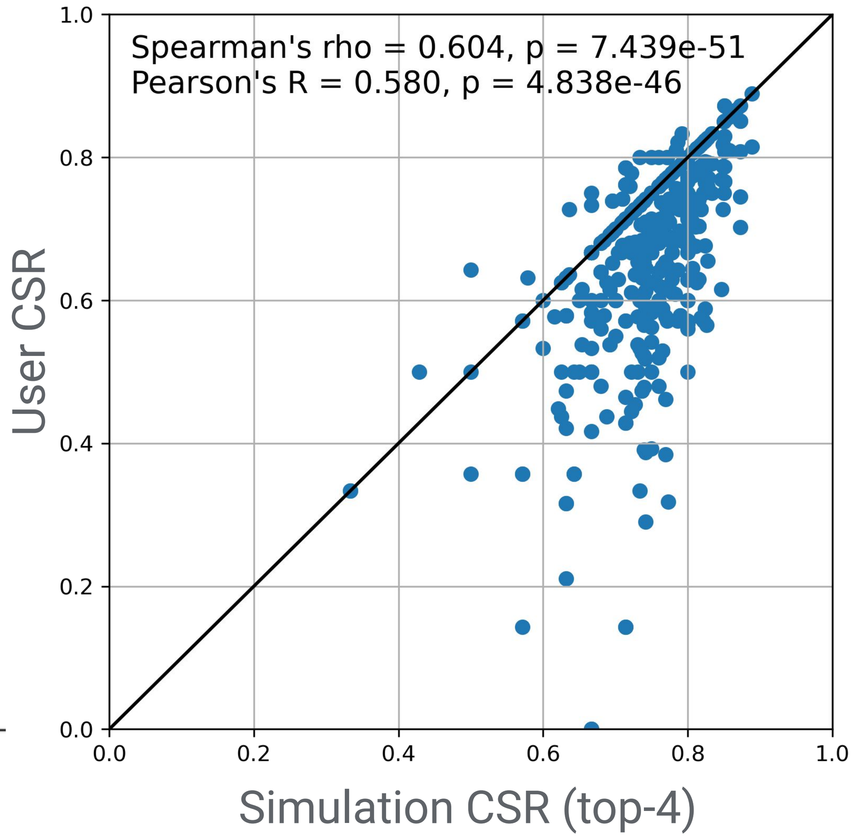
Goal: Understand to what extent users could use the potential of our interface and underlying LLM for motor saving.

Strategy: Simulate the most aggressive abbreviation (i.e., always prefer minimal input instead of appending more characters for safer decoding).

- **Step 1**: Start with the initials of each word.
- **Step 2**: If the target not in the candidates: append one more character to the first wrong word.
- **Step 3**: Repeat Step 2 until the target appears.
- **Step 4**: Get the final abbreviation.



CSR Correlation: user vs. simulation



Contributions Summary

- An **intuitive interface** and a **robust decoder** for seamless writing and editing of variable-length prefix-based abbreviations.
- A **user study** demonstrating a 60% reduction in motor efforts during handwriting, with competitive speed and accuracy.
- An **offline simulation** that quantifies the limit of LLM decoding capabilities for phrase abbreviations and examines how users' abbreviation behavior approached the upper bound of the LLM's abbreviation-expansion capability.