Improving FlexType: Ambiguous Text Input for Users with Visual Impairments

Dylan Gaines
Keith Vertanen
dcgaines@mtu.edu
vertanen@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

ABSTRACT

We present an improved version of the FlexType interface for nonvisual text input. FlexType enables nonvisual text input on mobile touchscreen devices by allowing users to select from a small number of character groups with gestures instead of targeting letters at specific screen locations. Based on an interview with users who are blind or low vision, we added a letter-entry mode to enable easier entry of difficult words such as proper nouns. We conducted a longitudinal study with users who are legally blind to compare FlexType to users' typical text input methods. While we found FlexType was significantly slower than Apple's onscreen Braille input, there was no significant difference in entry or error rate between FlexType and an onscreen keyboard with VoiceOver enabled. This was despite the participants having much more experience with the VoiceOver keyboard compared to FlexType. Overall, legally blind participants averaged 7.7 words per minute with a 7.0% character error rate after correction when writing with FlexType.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility technologies**; **Empirical studies in ubiquitous and mobile computing**; **Text input**; *Auditory feedback*; *Empirical studies in HCI*.

KEYWORDS

Nonvisual Text Entry, Accessibility, Human-Computer Interaction, Mobile Keyboard, User-Centered Design

ACM Reference Format:
Dylan Gaines and Keith Vertanen. 2024. Improving FlexType: Ambiguous Text Input for Users with Visual Impairments. In *The PErvasive Technologies Related to Assistive Environments (PETRA) conference (PETRA '24), June 26–28, 2024, Crete, Greece.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3652037.3652059

1 INTRODUCTION

Today's mobile devices often lack physical keyboards, opting instead for virtual keyboards that appear on a touchscreen. While this allows manufacturers to include larger screens or to reduce the size of their devices, it can create challenges for individuals who are blind or low vision (BLV). Without the haptic feedback afforded by physical keys, it can be difficult for users who are BLV to reliably target specific keys when typing on a mobile device. While in this work we focus on accessible text input for users who are BLV, sighted users may also wish to perform nonvisual text input in certain situations. For example, if users are trying to be discreet they may be entering text under a table, or if users are multitasking their visual attention may be directed elsewhere.

In our previous work we developed the FlexType text input interface to address this problem [7]. In FlexType, all of the letters and apostrophe are divided into four groups. Instead of selecting individual characters, users select the groups containing their intended characters by tapping anywhere on the screen with one to four fingers. After selecting the group for each letter in their intended word, users swipe to the right and the system *disambiguates* the input, determining the most likely matching word given the context. If the most likely word is not their intended word, users can then scroll through the *n-best list* of the *n* most likely words to find their target word.

Our previous study compared the longitudinal performance of FlexType with two different sets of groups optimized to reduce the number of disambiguation errors [7]. One set of groups was constrained to alphabetical order, where each group could only contain an unbroken sequence of characters. The other set of groups was unconstrained, where no restrictions were placed on the characters that each group could contain. We did not find a significant difference in the long-term performance of the groups, but we did find that initially users made significantly more errors with the unconstrained groups. Overall, we previously found that users entered text at 12.8 words per minute with a 1.9% character error rate. However, while the screen of the device used in the study was blanked during text entry, all of the participants in our previous study were sighted. We extend this work to make three primary contributions:

(1) An improved FlexType system that incorporates feedback from both a semi-structured interview with participants who are BLV and from users in the prior study [7].
(2) An input model that improves the disambiguation accuracy of FlexType and allows the system to correct errors in user input.
(3) A longitudinal evaluation with users who are BLV that compares FlexType to their typical onscreen text input method.

PETRA '24, June 26–28, 2024, Crete, Greece

ACM ISBN 979-8-4007-1760-4/24/06
https://doi.org/10.1145/3652037.3652059

2 RELATED WORK

FlexType [7] is not the first system to address nonvisual text input. The item selection technique used by Slide Rule [11] allows users to scan through a list of items by swiping their finger across the screen and select the item they are currently on by tapping a second finger in a different location. While the studies performed by Kane et al. did not focus on text input specifically, the authors noted that their system did enable text input using a Qwerty keyboard. A similar technique can be used with an onscreen keyboard using Apple's VoiceOver[1] and Android's TalkBack[2]. In VoiceOver or TalkBack, users can perform a double-tap gesture to enter a selected key or simply enter the selected key when they lift their finger, depending on their settings.

Several interfaces have leveraged the Braille alphabet to perform nonvisual text input. These types of interfaces rely on users designating which of the six dots are raised in the Braille encoding of each letter. This can be done using different interactions, such as two three-fingered taps to indicate which dots are raised on each side [2, 16], three two-fingered taps to indicate each row [12], or dragging gestures through each raised dot [13]. While the Braille-based interfaces can work well for users familiar with Braille, they require users to learn the corresponding mapping for each individual character and to use multiple gestures (or multiple hands). While FlexType users need to remember which characters are in each group, the overall number of gestures is much lower, and all gestures can be performed by a single action with a single hand.

As an alternative to typing altogether, users can also dictate their text and have it be interpreted by a speech recognition algorithm. While speech recognition was explored on mobile devices by Fischer et al. [6], it was several more years before its performance as a nonvisual text input method was investigated. Azenkot and Lee [1] conducted the first research on speech input with users who were BLV. They found that participants were able to use speech input to enter text at 19.5 words per minute but spent 80% of their time correcting errors.

Vertanen et al. [20] proposed a method where users would imagine a keyboard on the screen and type a sentence without any feedback until the sentence was complete. The system would then decode the entire sequence of taps into text. Users entered text at 23.3 words per minute while blindfolded, but with an 18.5% character error rate. Similarly, FlickType[3] allows users to enter text by tapping approximate character locations instead of definitively locating each key. Instead of entering a full sentence at once, FlickType users swipe to the right to signal to the system that they are finished entering a word. The system produces a best guess of the user's intended word from their tap locations. The user can also swipe through a list of suggested words if the system's best guess was incorrect.

Ambiguous keyboards, like FlexType, that place multiple characters on the same key have been used in the past. Before most commercial mobile phones had full Qwerty physical or touchscreen keyboards, they had nine-key (T9) ambiguous keyboards. Users would repeatedly press a key to iterate through the characters assigned to that key until they reached their desired character. Several studies have found this method to be slow, likely due to the need for multiple key presses for most characters [3, 10]. To remedy this, Tegic Communications [9] developed a method that would search a dictionary for words that matched the sequence of key presses a user entered and return the most likely matching word. Using this method, users only needed to press each key once per letter. More recently, ambiguous keyboards have been applied to smartwatches, where the screen size makes a full keyboard more challenging [4, 15]. TipText is a wearable text entry method that uses motion tracking or a conductive-printed film to detect touch locations on a user's fingertip [21]. In TipText, users entered text by touching their thumb to one of six ambiguous keys located on the tip of their index finger. TipText users entered text at an average of 12 words per minute.

3 INTERVIEWS WITH BLIND SMARTPHONE USERS

To identify potential areas for improvement for the FlexType interface, we conducted semi-structured interviews with twelve legally blind adults recruited from the National Federation of the Blind email list. Participants were selected based on the order that they responded to the advertisement. The interviews took approximately 30–60 minutes, and participants received a US$20 Amazon gift card as compensation. Participants ranged from 38 to 66 years old (mean 50). Six identified as female, five as male, and one did not identify with either gender. Five of the participants were completely blind, with one more having only minimal light perception. Eight participants reported being blind since birth, and all participants had been blind for a minimum of 19 years.

When reviewing the results of our interviews, we looked for common themes in the responses of our participants. We sought to identify experiences or opinions that many of our participants shared to help guide the direction of nonvisual text input research. We began by asking participants about their experiences with the text input methods that they currently use or have used in the past. While participants most commonly reported using speech recognition, all ten participants that used it reported poor recognition accuracy. Five of these participants noted that the accuracy was particularly poor in noisy environments, and six of the ten cited privacy concerns regarding using speech recognition in public places. One participant that used onscreen Braille input also had privacy concerns regarding their screen facing away from them during text input. This highlights the need for an alternative method, such as FlexType.

Next, we described the FlexType interface to participants and asked if they would be interested in using it. Participants had mixed enthusiasm about FlexType, but eight of the twelve were interested in trying it, and only two were firmly against the idea. The chief concern that participants voiced was the steep learning curve involved with learning the new method and memorizing the groups of characters.

Participants were also concerned about the accuracy of the algorithm determining their intended word. In the prior study [7], if a user's intended word was not one of the six most likely words

[1]https://support.apple.com/guide/iphone/use-the-onscreen-keyboard-iph3e2e3d1d/ios
[2]https://www.android.com/accessibility/vision/
[3]https://www.flicktype.com/

for the entered tap sequence, there was not a way for the user to type that word. We most commonly observed this scenario for intended words such as proper names or abbreviations. We asked participants how they would foresee entering these types of words using FlexType. Four participants recommended a personal user dictionary, where they could predefine a list of proper names to add to the system's vocabulary, and three requested the ability to enter text one letter at a time when needed.

Many keyboard interfaces commonly include word suggestions to help users input text more quickly. These are often displayed along the top of an onscreen keyboard. Users can select from likely options at any point, even before finishing a word. While the original version of FlexType did not include word suggestions prior to each word's completion, we considered adding this feature to the interface in this work. When we asked the interview participants about their usage of word suggestions, seven of the participants either did not use word suggestions at all or had the option disabled, one reported having used them very rarely, and the other four said they actively used them. All four participants that actively used word suggestions said that they needed to explore them with their screen reader and then select them like they would a key with VoiceOver. Participants that did not use word suggestions often said that they were disruptive to use, and that it was easier or faster for them to just finish typing a word. Based on this mixed feedback, we decided against adding word suggestions to the new version of FlexType.

4 SYSTEM DESCRIPTION

In this study, participants completed each session remotely on their own device. Since the vast majority of our interview participants used an iPhone as their primary mobile device, we ported the Android version of FlexType used in the previous study [7] to iOS (the iPhone operating system). All participants used a smartphone device running at least iOS 16.0.

4.1 Available Gestures

The primary functions of the interface remained the same as the original FlexType, with users tapping anywhere on the screen with one to four fingers at the same time to select from the four character groups. Since the previous study [7] did not find a significant difference in entry or error rate between the Constrained and Unconstrained groups and, as recommended by our interview participants, we wanted to lower the barrier to entry, we selected the Constrained FlexType groups for our interface:

$$
\begin{aligned}
&1: (a, b, c, d, e),\\
&2: (f, g, h, i, j, k, l, m),\\
&3: (n, o, p, q, r),\\
&4: (s, t, u, v, w, x, y, z, ')
\end{aligned}
$$

After entering the full sequence of taps for each word, participants swiped right to use the VelociTap decoder [19] to disambiguate the sequence. Mimicking the previous study, participants were able to swipe up to navigate to the next word in the list of six recognition results and swipe down to return to the previous word [7]. A sound effect played to indicate if a participant attempted to navigate past the end of the list. Participants were able to swipe left with one finger to backspace a character and with two fingers to backspace a word. The version of FlexType used in this work also added a three-finger left swipe gesture to clear the entire text field. A long press (over 600 ms) with one finger would cause the interface to read the prompt again as well as the text the user had entered so far. A long press with two fingers would cause the interface to speak the available gestures. During text entry, FlexType provided visual feedback of previously recognized words and the current pending ambiguous taps by displaying them in the center of the screen in large high-contrast text (Figure 1).

Figure 1: Example of the FlexType interface showing a user who is entering the phrase "please call". The user has completely entered the word "please", which is displayed in large high-contrast text. The user has selected the groups for each letter in "call", but has not yet swiped right to perform recognition. The unrecognized taps are displayed using the selected group numbers "1122".

4.2 Decoder Model

The decoder that performed disambiguation ran on a remote server, and was queried for recognition results through a REST API. The decoder used a 4-gram word model and 12-gram character model to score word hypotheses. To determine the best decoder configuration to use, we tested and compared decoder models with different sets of parameters. The baseline decoder model had parameters for the mixture weights between the language models, an out-of-vocabulary penalty (assessed to hypotheses not in a list of 100K frequent words that also appear in human-edited dictionaries), and a beam width controlling how the search was pruned. In the previous study [7], the decoder restricted its search to only words that exactly matched the group sequence entered by the user (e.g., "cat" would be an acceptable hypothesis if and only if the group sequence was 1–1–4, since 'A' and 'C' are in group 1, and and 'T' is in group 4). To attempt to correct user errors in this study, we tested decoder models which added to the baseline model different tunable parameters that allowed the decoder to insert, delete, and/or substitute characters.

We used the data from the previous FlexType study [7] to tune the decoder parameters and compare the different models. We compared each candidate model using leave-one-out cross validation, withholding one of the eight participants' data in each training fold to evaluate the model's performance on an unseen participant. More details of the tuning process can be found in [8] (Chapter 7).

The best model we found allowed the decoder to produce hypotheses that substituted characters from different groups than the user had input. For every substitution in a given hypothesis, a penalty was assessed to the likelihood of that hypothesis. The magnitude of the penalty was determined by a single tunable parameter n, set to the exponent of the distance between the input group and the substituted group. Because of this, a group that was further from the input incurred a larger penalty. For example, a hypothesis with a character from group 3 where the user had input group 4 would incur a penalty of n, since it was a difference of one group. A hypothesis that substituted a character from group 1 instead would incur a penalty of n^3, since it was a substitution from three groups away. Since the decoder was searching for the most likely hypotheses, errors in the input would make it more difficult for the decoder to produce the desired text, but not impossible if the desired text was likely under the language model.

To avoid potentially unexpected behavior while the participants were learning the technique and the groups (e.g. the decoder performing a substitution during recognition), the first three sessions of our user study used the baseline model, which required disambiguation results to exactly match the user input. The remainder of the sessions used the best model that allowed substitutions.

4.3 Entry Modes

By default, FlexType began each entry task in *word entry mode*, which is the mode that we have described thus far. With a two-finger swipe up, participants were able to toggle to *letter entry mode*, the other new interface feature we introduce to FlexType in this work. This mode allowed participants to enter text a single letter at a time to provide a deterministic input mode that was not influenced by the decoder. In letter entry mode, each group selection would enter and speak the character in the middle of the selected group (e.g. 'c' in group 1). Participants could then swipe up or down to iterate forwards or backwards through the characters in the group, the same way they could to explore the n-best list in word entry mode. In letter entry mode, a right swipe would enter a space and a two-finger swipe up would toggle back to word entry mode.

We opted to implement letter entry mode as opposed to increasing the size of the n-best list produced by the decoder. Our main reasoning was that increasing the size of the n-best list would still not be guaranteed to solve the problem, and it would lead to participants spending more time exploring deeper into the list. In the results of the previous study [7], 89% of correctly entered words were the first result returned by the decoder. The target word was in the first half of the n-best list in 98.6% of cases, and the target word was not in the n-best list at all only 0.7% of the time. The fourth, fifth, and sixth positions contained only 0.4%, 0.2%, and 0.1% of entered words, respectively. Even assuming that subsequent positions in an expanded n-best list would maintain the 0.1% of entered words seen in the sixth position (which is unlikely given the trend of the data), we would need to double the size of the n-best list to include all of the words seen by the users in the previous study. Since additional out-of-vocabulary words such as proper nouns specific to individual users' lives are not in our phrase lists, it is unlikely that even an expanded n-best list would allow users to enter any word.

A personal user dictionary, as suggested by interview participants, would require the user to predefine words that might not be in the decoder's vocabulary. Proper names that are new to the user (e.g. someone they just met or a new restaurant they want to try) might not be included yet, and the user would not be able to type them until they were defined. While not included in this study, a potential implementation of this could automatically add words to a user's personal dictionary that they type using letter entry mode.

5 PROCEDURE

Prior to the study, we employed a third party accessibility expert in the blind/low vision field to test the FlexType app. We used this expert's feedback to refine the length of each session and to make the app more accessible for users with visual impairments.

We made the study app available to participants through Apple's TestFlight, a system designed for developers to test preliminary versions of their applications. This allowed us to restrict access to only the participants that were recruited for the study.

Once participants downloaded the app, we presented them with an informed consent form, followed by an introductory questionnaire. At the beginning of each session, we showed participants an instruction page that included a description of what was new in that session, followed by a reminder of the available gestures introduced in previous sessions. We instructed participants to hold the device in portrait orientation in one hand and use the other hand to tap the screen. We felt this would best simulate a mobile text input scenario in which the user could not place the device on a surface (e.g. on public transit).

Each participant completed eight sessions designed to progressively get harder and introduce interface features. The first four sessions were designed as practice, while we used the final four to compare the FlexType interface to each participant's typical text entry method.

- **Session 1** — Participants received single-character prompts and were instructed to tap with the number of fingers corresponding to that character's group. After each tap, the characters in the selected group were read, followed by feedback on whether the selection was correct or not. Participants had to get each character correct before proceeding to the next target character. A long press in this session would read the target character as well as its word from the NATO phonetic alphabet to help distinguish between similar sounding characters. This feature was suggested by our accessibility expert. Participants entered each character (A–Z and apostrophe) four times, in a randomized order.
- **Session 2** — Participants received 54 single-word prompts from a list of common words. These prompts were chosen to

ensure that each participant encountered each character at least three times. The prompts in this session only contained words that could be written without requiring participants to explore the n-best list (i.e. participants did not need to swipe up or down after recognition). Due to a bug, one word ("they'd") did require use of the n-best list. We excluded this prompt from analysis.

- **Session 3** — Participants received 20 phrase prompts that contained no more than four words, with each word being no longer than six characters. The prompts were not pruned to remove words not appearing as the first result in the n-best list, but they were pruned to remove words that did not appear at all in the n-best list. (i.e. participants could always enter the intended word if they did not make errors, but they may have needed to swipe up or down after recognition). The first prompt each participant received contained a word that required exploring the n-best list. All phrase prompts were drawn from the memorable subset of the Enron mobile data set [18].
- **Session 4** — Participants received 20 phrase prompts that were pruned to contain no more than four words, with each word being no longer than six characters. The prompts were no longer pruned to remove words not appearing in the n-best list, and the first prompt each participant received contained a word that would not appear in the n-best list. This allowed participants to practice using letter entry mode immediately after the instruction page where it was described.
- **Sessions 5–8** — Participants received 20 phrase prompts in each of two conditions, FlexType and Baseline. The FlexType condition used the same entry process as the previous sessions, and the Baseline condition instructed participants to use their typical onscreen text input method. We instructed participants to use the same input method in the Baseline condition during each session. The order in which participants completed the conditions alternated between sessions and between participants. For example, Participant 1 completed the FlexType condition first in sessions 5 and 7 and second in sessions 6 and 8, while Participant 2 completed the FlexType condition second in sessions 5 and 7, and first in sessions 6 and 8. Prompts were pruned to contain no more than six words to help participants remember them, but no limits were placed on word length or whether or words appeared in the n-best list.

Based on feedback from participants in the previous study [7], the corresponding character group was only read after each tap in the first two sessions. In all subsequent sessions, a key tap sound was played instead.

Prior to entry, FlexType displayed the target text and read it via text-to-speech. Users were instructed to tap anywhere to continue and begin entry. In all sessions after the first, participants swiped down with two fingers to indicate when they were finished entering a prompt. They were then presented with their accuracy and speed for that prompt before they tapped to continue to the next prompt. Participants completed a questionnaire at the end of each of the first four sessions, and after each condition in the final four sessions.

In the portion of each session using FlexType, the app asked participants to disable VoiceOver (if the system detected it was enabled) prior to receiving the first prompt. This was necessary since if VoiceOver was enabled it would intercept screen touch events and prevent FlexType from interpreting them. We designed the app to provide its own audio feedback in instances where VoiceOver needed to be disabled. We discuss some of the drawbacks of this design in Section 7.

Participants were instructed to complete only one session in a single day and to have at most one day off between sessions. If participants enabled the permissions, they received a notification both 24 and 48 hours after completing a session if they had not yet begun the next session. They also received a notification if 15 minutes had passed mid-session without user activity.

6 RESULTS

We recruited a total of 25 participants from the National Federation for the Blind email list, none of whom participated in our interview. Of these participants, four never downloaded the app for the study and five failed to complete all study sessions.

From the 16 complete participants, one participant displayed signs of having significant difficulty with the word-at-a-time entry technique utilized by FlexType. The majority of this participant's input in the training sessions consisted of disambiguating a single character at a time instead of a full word, leaving spaces between each character. Because this was a remote and asynchronous study, we were unable to correct this participant's misunderstanding of the input technique. This participant used exclusively letter entry mode for all input in the FlexType conditions of sessions 5–8. We excluded this participant from our analysis.

Another participant, although instructed to use their typical onscreen input method, used a wireless physical keyboard for the Baseline condition. While some of our interview participants used wireless keyboards on occasion, most remarked that carrying the extra equipment with them was too much of a burden. Since the goal of this research was to improve onscreen text input for instances where a physical keyboard is not available, we also excluded this participant from analysis.

The 14 remaining participants were aged between 21 and 59 (mean 39). Four identified as male, and ten as female. Seven participants reported being blind since birth, and all participants had been legally blind for more than five years. Five participants were completely blind, and five more had only light perception. All participants rated the statement "I am a fluent speaker of English" as a 7 on a 7-point Likert scale, where 7 represented "strongly agree". When asked to rate the statement "I frequently enter text on a mobile phone" on the same scale, participants responded with an average of 6.8.

6.1 Entry Rate

We measured participants' entry rate in words per minute (WPM), considering every five characters, including spaces, to be a word. As shown in Figure 2, entry rate varied considerably between the participants. Some participants showed great improvement, with Participant 1 improving from 5.4 WPM in session 2 to 12.8 WPM in session 8, and Participant 7 improving from 10.1 WPM to 17.0 WPM.

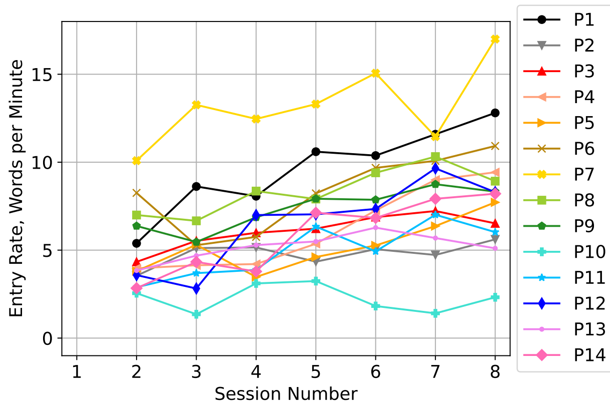

Figure 2: Entry rates of each participant on FlexType throughout the study. Session 1 entry rates are not reported since participants only entered single-character prompts.

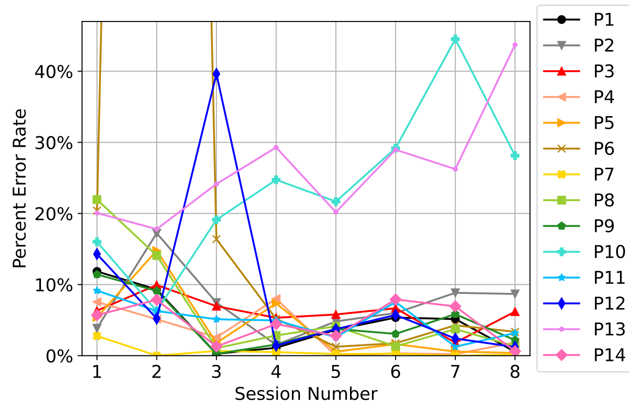

Figure 4: The error rate of each participant in the FlexType condition throughout the user study. Session 1 results depict tap error rate, and subsequent sessions depict character error rate. In session 2, P6 had a CER of 386%.

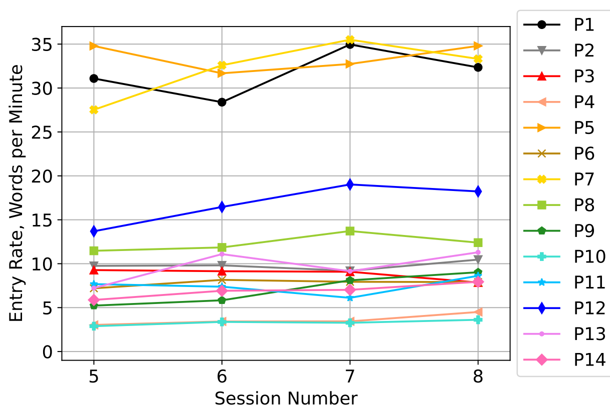

Figure 3: Entry rates of each participant with their Baseline text input method. Participants only used this method in sessions 5 and later.

Other participants did not see the same improvement, for example, Participant 10 regressed from 2.6 WPM to 2.3 WPM over the course of their sessions. Overall, participants averaged 7.7 WPM over their final four sessions. This had quite a bit of variability, with a range from 2.2 WPM to 14.2 WPM and a standard deviation of 2.8 WPM. Overall, participants did appear to get faster. The mean entry rate in session 5 was 7.0 WPM, which increased every session thereafter, reaching 8.4 WPM in session 8.

When comparing participants' results with their typical text entry methods, we chose to look at each baseline method independently. Of our 14 participants, 11 used a Qwerty keyboard with VoiceOver in their Baseline condition. In the final four sessions, these 11 participants had a mean FlexType entry rate of 6.7 ± a standard deviation of 2.1 WPM, and a mean Baseline entry rate of 10.0 ± 8.2 WPM. Since a Shapiro-Wilk test showed a violation of the normality assumption (W = 0.66, p < 0.001), we used a Wilcoxon signed-rank test. This test did not show that this difference was significant (V = 18, p = 0.21, moderate effect size: r = 0.40).

The other three participants (P1, P7, P11) used Braille Screen Input as their typical text input method. These participants had a mean FlexType entry rate of 11.2 ± 3.1 WPM, and a mean Baseline entry rate of 26.9 ± 8.7 WPM. A Shapiro-Wilk test showed no violations of the normality assumption (W = 0.89, p = 0.36) and a dependent t-test found the Baseline was significantly faster ($t(2)$ = -4.44, p = 0.047, large effect size: Cohen's d = 1.16).

Each participant's Baseline entry rates for each session can be seen in Figure 3. While most participants were fairly static throughout the sessions, some (e.g. P7 and P12) seemed to get slightly faster in later sessions. We hypothesize that this was due to the participants gaining familiarity with the study app and the process of submitting their text, as opposed to improvement in their use of their typical text entry method.

6.2 Error Rate

To evaluate our participants' error rates in session 1, we used the error rate of their taps, since they were only being asked to select the group containing a character, and they were not entering text. For example, if a participant required three attempts to select the correct group, this would yield a 67% error rate for that prompt.

For the remainder of the sessions, we used Character Error Rate (CER). We computed the number of insertions, deletions, and substitutions required to transform the participant's final text (after any corrections) to the reference text (the prompt), divided by the length of the reference text. Since it was possible for the length of the input to exceed the length of the prompt, it was possible to obtain a CER that exceeded 100%.

As shown in Figure 4, we did observe a CER over 100% for Participant 6, who had a misunderstanding about how FlexType worked in session 2, which led to an average CER of nearly 386%. The participant was tapping each group multiple times to get to the desired character (e.g. tapping with two fingers five times to get to 'J', the fifth letter in group two). This led to excessively long tap sequences that the decoder then failed to disambiguate since there were no words matching those sequences. The participant reached out to us following that session and we were able to clarify that the disambiguation process would choose the letters from the groups after the word was completed. Participant 6 went on to average 2.7% CER with FlexType over the final four sessions.

Overall, participants had slightly higher error rates in the first couple sessions that then decreased as they became more familiar with the text entry technique. Across the final four sessions, participants had an average FlexType CER of 7.0%, though this was inflated by Participants 10 and 13 who had CERs of 30.9% and 29.8%, respectively, in their final four sessions. We will discuss more about why these participants may have struggled in Section 8. More resistant to these outliers, the median CER of participants in their final four sessions was 3.7%.

The differences in character error rate between the FlexType and Baseline conditions for the 11 participants that used a Qwerty keyboard with VoiceOver violated the normality assumption (W = 0.79, p = 0.006). While these participants had a mean FlexType CER of 8.3% ± 11.0%, and a mean Baseline CER of 4.0% ± 4.1%, a Wilcoxon signed-rank test did not find this difference to be significant (V = 51, p = 0.12, moderate effect size: r = 0.48).

For the three Braille Screen Input users, we found a mean FlexType CER of 2.4% ± 1.9%, and a mean Baseline CER of 0.5% ± 0.1%. The differences did not violate normality (W = 0.81, p = 0.15), but a dependent t-test did not show a significant difference ($t(2)$ = 1.77, p = 0.22, small effect size: Cohen's d = 0.39).

6.3 Error Correction and Prevention

For each session, we calculated the number of characters each participant backspaced per character in their final text. This provides a measure of how much correction a participant had to perform. For this metric, we used the total characters deleted, summed across all methods of backspacing (single character, full word, full input). As shown in Figure 5, most participants had a fairly static number of backspaces per character (BPC) throughout their sessions. The exception to this was, again, Participant 10, who backspaced quite frequently in the later sessions, and over 6 times per final character in session 7. For the final four sessions, participants had a mean BPC of 0.39 ± 0.66, with a median of 0.21 BPC.

In the previous study [7], we saw a relatively high BPC in session 1, followed by a fairly constant and low BPC in the remaining sessions. In this study, we adjusted session 1 to have participants try again if they selected the incorrect group without the need (or ability) to backspace. Because of this, we do not have a measure of participants' BPC in session 1. However, if we assume that participants backspaced each incorrect tap in session 1, this would yield a theoretical mean BPC of 0.42 ± 0.46 for that session, with a median of 0.25 BPC. This was fairly similar to the BPC performed in the final four sessions, whereas the previous study showed a decrease in BPC after the first session.

Of the characters backspaced in the final four sessions, 76% were done a single character at a time, while 23% were deleted using the word-at-a-time feature. Only 1% of characters backspaced were the result of participants clearing the entire entry field. However,

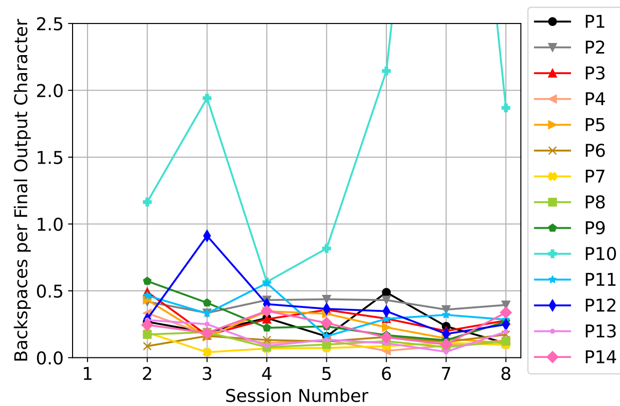

Figure 5: Each participant's backspaces per final output character in the FlexType condition. This is not reported for session 1 since participants were not able to backspace. In session 7, Participant 10 had a BPC of 6.1.

this was affected drastically by Participant 10 since they performed such a large proportion of the backspaces. Excluding Participant 10, the remaining participants deleted 57% of characters one at a time, 43% as part of a word, and 0.6% by clearing the entry field. Participant 10 deleted 98% of character one at a time, 0.4% as part of a word, and just over 1% by clearing the entry field.

In the Baseline condition, since participants used their typical text input methods that we had not developed, we were not able to log input events and could only access the final text for each prompt. Because of this, we were not able to measure any sort of corrective action, such as backspaces per character, for the Baseline condition.

In total, participants' final text contained 4394 words in the final four sessions. This made up only 81% of the 5430 total times they queried the decoder for word recognition; the remainder were later deleted by the participants. Participants explored the n-best list in a total of 16% of the words they entered, including words that were later deleted. This exploration was comprised of 2092 up swipes (to navigate to the next word in the n-best list) and 356 down swipes (to navigate to the previous word), for a total of 0.56 swipes per word in their final output text. Participants performed an average of 2.9 swipes per word when considering only the words where they explored the n-best list, which shows that on average, participants were exploring more than half of the list if they explored it at all.

To prevent errors by the decoder, participants could switch to letter entry mode and enter text one letter at a time. Participant 2 used this feature quite often, switching it on over 13 times per session in their final four sessions. The other participants, however, only enabled letter entry mode an average of 1.2 times per session in their final four sessions, with two participants never using it. Three participants did not use the feature at all in session 4, where the first prompt was curated to have a word that would not appear in the n-best list. We believe that this may have been caused by participants skipping over the instructions that explained the feature, or by participants finding a word in the n-best list that they felt was close enough that it did not warrant the additional time to correct.

7 LIMITATIONS

One limitation of our interface was that FlexType was not integrated with VoiceOver. Since VoiceOver, if enabled, intercepts touchscreen events and processes them itself, participants needed to disable VoiceOver in the FlexType condition. While we developed the app to provide its own audio feedback, both during entry and to read each prompt, Participant 3 noted after a Baseline condition that "Being able to use VoiceOver enabled me to actually read the words I was supposed to be typing individually and go character by character, so I was able to know how certain things were spelled". This was not available in the FlexType condition, and may have contributed to higher error rates when compared to participants' Baseline methods.

Further limitations of this study arose from it being conducted entirely remotely. Since participants were using their own devices for the study, they had a variety of devices and operating system versions. This led to some bugs that we did not encounter during testing. For example, Participant 10 noted that whenever they tried to switch to letter entry mode, the app would crash. Participant 13 cited frequent crashes that we determined were a result of the web server running the decoder API crashing. In an in-person study, we would have been able to detect and remedy these issues more immediately. Participant 6 also noted crashes that occurred when they tapped with three fingers, since it conflicted with gestures used by their screen magnification software. The remote study also made it more difficult to clarify any misunderstandings of the instructions. While some participants reached out via email to ask questions, others did not, which in the case of some participants led to poorer performance. An in-person study with the ability to standardize the device would have allowed participants a smoother experience with FlexType. However, it would have been more difficult to recruit a large and diverse pool of participants.

Another difficulty of the remote study was enforcing our desired session pacing. While six participants followed our pacing instructions, six other participants had a large time gap within a single session. Participant 11 had an overnight 14-hour gap partway through session 4, and Participant 14 had a one-day gap between two prompts in session 2. Participant 6 encountered a bug within the final session that took us a few days to fix. On several occasions, Participant 13 took multiple days to complete a single session, and Participant 5 often had several days to a week between the two conditions of a single session. Participant 12 had a three-day gap partway through session 3, and a two-week hiatus between sessions 7 and 8. Participant 2 had a 4-day gap between their first and second session, and Participant 8 completed all four of their final sessions within a 14-hour total time span. While some of these occurrences were minor and we have no reason to believe they impacted the final results, Participant 8's results may have been affected by the fatigue of doing so many sessions in quick succession. Additionally, the large time gaps we observed for Participants 5, 12, and 13 may have reduced participants' ability to remember interface features or character groupings.

While we recruited 25 total participants, we only had 14 for our analysis. This led to limited power for our statistical tests, especially for the users of Braille Screen Input (BSI), where we only had a sample size of three. While we did recruit more participants that used BSI, not all of them finished the study. This also created a slight imbalance in the ordering of conditions. Since all three participants that used BSI had odd participant numbers, they completed the FlexType condition first in session 5. Since we swapped the order of conditions in alternating sessions (odd numbered participants completed the Baseline condition first in sessions 6 and 8), we believe the effect of this order imbalance on the final data was minimal.

8 DISCUSSION

In the FlexType condition, Participant 10 had the lowest entry rate and both error rates and BPC that were quite elevated. In post-session comments, Participant 10 stated that they had trouble getting their fourth finger down to select group 4, and that once the feedback on which group had been selected went away, it was difficult for them to know if it had been recognized properly. In this study, we removed the audio feedback that read the selected group after each tap in session 3 and later. This decision was based on feedback from participants in the previous study [7] who said they found the audio feedback annoying in later sessions once they were familiar with the groups.

Participant 13 also showed an elevated error rate across most of their FlexType sessions. Post-session feedback from Participant 13 also referenced difficulty with the three- and four-finger taps due to a small device size. The experiences of Participants 10 and 13 suggest that the group audio feedback should be a configurable setting for users. A possible compromise option would be to read the selected group number after each tap, which was suggested by Participant 1, or to play some other sound effect that is unique to each group. While some participants in our previous study [7] noted difficulty tapping with four fingers (e.g. one participant stated "tapping with all 4 fingers is difficult when tapping fast"), none expressed so much difficulty that we felt it necessitated a change. Seeing Participant 10's and 13's consistent comments that they struggled to select group four, it is worth considering a reduction to three groups or devising an alternate way to select group four. While not as ergonomic, it may have also been beneficial for participants to hold the phone in landscape orientation instead of portrait.

Based on the feedback from our interviews, we chose not to include word suggestions prior to the completion of each word. However, after their Baseline condition in session 5, Participant 8 commented "some keyboards have autocorrect, and some keyboards have suggested words which could lead to shortcuts". If some participants were using these features with their typical text entry method, that could have led to increased performance in the Baseline condition. While we asked participants to report the text entry method they used for the Baseline condition, we did not ask about their use of word suggestions or autocorrect. In future development of FlexType, it could be beneficial to implement word suggestions. One possible implementation would be to play the most likely prediction after each tap and allow users to swipe right to select it. This would allow users to know what result they will get when they swipe right and potentially increase their entry rate by eliminating keystrokes. However, this has the potential to confuse users if their intended word has a similar starting group sequence to the words being read, without necessarily having similar letters (e.g. "help" and "fair" have the same tap sequence without sharing any letters).

While we were hopeful that FlexType might outperform typical onscreen text input methods, this was not the case. When comparing our participants' performance on FlexType to their baseline text entry methods, the only significant difference we found was that Braille Screen Input users were faster with their typical method. This was supported by participant comments, with Participant 11 remarking after the FlexType condition in session 6, "after using my normal text input method, I found this method to be especially slow". However, compared to their typical text input methods, participants had very little practice with FlexType. After the final session, Participant 11 was hopeful for the future of FlexType, stating, "I believe right now that FlexType is slower than my regular text input method, but, I am sure with improvements it can become faster than Braille screen input." Conversely, Participant 1 stated they did not think FlexType would ever be faster than Braille screen input for them, but that it would be great for people who did not know Braille.

While we found no significant difference between a Qwerty keyboard with VoiceOver and FlexType, the majority of subjective user feedback seemed to favor FlexType. Of the 11 participants that used a Qwerty keyboard with VoiceOver as their Baseline text input method, seven expressed that they wanted to learn more about the development of FlexType in the future. In the final questionnaire, Participant 8 commented, "I really hope that this entry method becomes widely available because I am honestly sad that I don't get to use it anymore. I think that a lot of blind people would appreciate this method since it is so much easier than using the typical typing interface." Participant 4 preferred FlexType to their typical method, stating, "When I had to go back to entering on the QWERTY keyboard on screen, it felt slow and clumsy." Negative comments about FlexType were mostly related to difficulty getting used to the technique, bugs with the app for the study, and issues with tap recognition. Participant 10 suggested the possibility of using VoiceOver to select the groups, which would alleviate the issue some encountered selecting the four-finger gesture.

9 CONCLUSION

In this work, we improved the FlexType nonvisual text input method based on feedback from users who are BLV. We used character groups that were constrained to alphabetical order to lower the barrier to entry, and we added letter entry mode to allow users to type any word, even if it was not recognized by the decoder. We used the data collected in our prior study [7] to tune the decoder to model user input errors in its disambiguation process. We conducted a longitudinal study to compare legally blind users' performance using FlexType to their typical text input methods. Three users input text using Braille Screen Input and had an average entry rate of 26.9 words per minute. These same users averaged 11.2 words per minute with FlexType, which was significantly slower. Eleven users input text using a Qwerty keyboard with VoiceOver at 10.0 words per minute, which was not significantly different from those users' FlexType entry rates of 6.7 words per minute. Feedback from some of the VoiceOver users stated that it felt easier and faster to use FlexType compared to their typical interface. Combining both groups, the 14 legally blind users entered text at an average of 7.7 words per minute using FlexType.

In this work we used the VelociTap decoder [19] to disambiguate sequences of character groups into words. For our user study, we performed disambiguation on a remote server, which could lead to latency or privacy concerns for a real-world input method. Further, VelociTap uses n-gram language models to make predictions based on the frequencies of character and word sequences in the models' training data. Recently, large language models (LLMs) have been shown to provide state-of-the-art performance for many natural language processing tasks [5, 14, 17, 22]. Future work could fine-tune a large language model, of an appropriate size to be able to run locally on users' devices, to disambiguate sequences of character groups into words. This could be done either after the completion of the word as we did here, or by producing likely word completions based on the first few groups selections in a word (or both).

Even though this was a longitudinal study, users only had about four hours of training before we began our evaluation sessions. Given that users reported entering text with their baseline methods frequently, our evaluation may not show FlexType's full potential. In future work, it would be interesting to evaluate how users improve with FlexType with continued practice. This could be done by releasing FlexType as a standalone input method that users could utilize for real-world text input. This would require additional consideration on how to enter characters such as capital letters, numbers, and punctuation, but it would allow us to explore how user performance evolves with long-term use. The subjective feedback from the participants in our study and their desire to continue using FlexType suggests that such a further evaluation could be promising.

ACKNOWLEDGMENTS

This work was supported by NSF IIS-1909248 and by NSF Graduate Research Fellowship 2034833. We would also like to thank the participants who took part in our studies and the National Federation of the Blind for advertising our studies.

REFERENCES

[1] Shiri Azenkot and Nicole B. Lee. 2013. Exploring the Use of Speech Input by Blind People on Mobile Devices. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility* (Bellevue, Washington) *(ASSETS '13)*. Association for Computing Machinery, New York, NY, USA, Article 11, 8 pages. https://doi.org/10.1145/2513383.2513440

[2] Shiri Azenkot, Jacob O. Wobbrock, Sanjana Prasain, and Richard E. Ladner. 2012. Input Finger Detection for Nonvisual Touch Screen Text Entry in Perkinput. In *Proceedings of Graphics Interface 2012* (Toronto, Ontario, Canada) *(GI '12)*. Canadian Information Processing Society, CAN, 121–129.

[3] Melinda M. Cerney, Brian D. Mila, and Lewis C. Hill. 2004. Comparison of Mobile Text Entry Methods. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 48, 5 (2004), 778–782. https://doi.org/10.1177/154193120404800508

[4] Mattia De Rosa, Vittorio Fuccella, Gennaro Costagliola, Giuseppe Adinolfi, Giovanni Ciampi, Antonio Corsuto, and Donato Di Sapia. 2020. T18: an ambiguous keyboard layout for smartwatches. In *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*. Institute of Electrical and Electronics Engineers, New York, NY, USA, 1–4. https://doi.org/10.1109/ICHMS49158.2020.9209483

[5] Jacob Devlin, Chang Ming-Wei, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

arXiv 1810.04805v2.

[6] Arnout R. H. Fischer, Kathleen J. Price, and Andrew Sears. 2005. Speech-Based Text Entry for Mobile Handheld Devices: An Analysis of Efficacy and Error Correction Techniques for Server-Based Solutions. *International journal of human-computer interaction* 19, 3 (2005), 279–304.

[7] Dylan Gaines, Mackenzie M Baker, and Keith Vertanen. 2023. FlexType: Flexible Text Input with a Small Set of Input Gestures. In *Proceedings of the 28th International Conference on Intelligent User Interfaces* (Sydney, NSW, Australia) *(IUI '23)*. Association for Computing Machinery, New York, NY, USA, 584–594. https://doi.org/10.1145/3581641.3584077

[8] Dylan C Gaines. 2023. *An Ambiguous Technique for Nonvisual Text Entry.* Ph. D. Dissertation. Michigan Technological University. https://doi.org/10.37099/mtu.dc.etdr/1667

[9] Tegic Communications Inc. U.S. Patent 5818437, Oct. 1998. Reduced Keyboard Disambiguating Computer.

[10] Christina L. James and Kelly M. Reischel. 2001. Text Input for Mobile Devices: Comparing Model Prediction to Actual Performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seattle, Washington, USA) *(CHI '01)*. Association for Computing Machinery, New York, NY, USA, 365–371. https://doi.org/10.1145/365024.365300

[11] Shaun K. Kane, Jeffrey P. Bigham, and Jacob O. Wobbrock. 2008. Slide Rule: Making Mobile Touch Screens Accessible to Blind People Using Multi-Touch Interaction Techniques. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility* (Halifax, Nova Scotia, Canada) *(Assets '08)*. Association for Computing Machinery, New York, NY, USA, 73–80. https://doi.org/10.1145/1414471.1414487

[12] Sergio Mascetti, Cristian Bernareggi, and Matteo Belotti. 2012. TypeInBraille: Quick Eyes-Free Typing on Smartphones. In *Computers Helping People with Special Needs*, Klaus Miesenberger, Arthur Karshmer, Petr Penaz, and Wolfgang Zagler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 615–622. https://doi.org/10.1007/978-3-642-31534-3_90

[13] Joao Oliveira, Tiago Guerreiro, Hugo Nicolau, Joaquim Jorge, and Daniel Gonçalves. 2011. BrailleType: Unleashing Braille over Touch Screen Mobile Phones. In *Proceedings of the 13th IFIP TC 13 International Conference on Human-Computer Interaction - Volume Part I* (Lisbon, Portugal) *(INTERACT'11)*. Springer-Verlag, Berlin, Heidelberg, 100–107. https://doi.org/10.1007/978-3-642-23774-4_10

[14] OpenAI. 2023. GPT-4 Technical Report. arXiv 2303.08774.

[15] Ryan Qin, Suwen Zhu, Yu-Hao Lin, Yu-Jung Ko, and Xiaojun Bi. 2018. Optimal-T9: An Optimized T9-like Keyboard for Small Touchscreen Devices. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (Tokyo, Japan) *(ISS '18)*. Association for Computing Machinery, New York, NY, USA, 137–146. https://doi.org/10.1145/3279778.3279786

[16] Caleb Southern, James Clawson, Brian Frey, Gregory Abowd, and Mario Romero. 2012. An Evaluation of BrailleTouch: Mobile Touchscreen Text Entry for the Visually Impaired. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (San Francisco, California, USA) *(MobileHCI '12)*. Association for Computing Machinery, New York, NY, USA, 317–326. https://doi.org/10.1145/2371574.2371623

[17] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv 2302.13971.

[18] 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* (Stockholm, Sweden) *(MobileHCI '11)*. Association for Computing Machinery, New York, NY, USA, 295–298. https://doi.org/10.1145/2037373.2037418

[19] 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. In *CHI '15: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seoul, Korea). Association for Computing Machinery, New York, NY, USA, 659–668. https://doi.org/10.1145/2702123.2702135

[20] Keith Vertanen, Haythem Memmi, and Per Ola Kristensson. 2013. The Feasibility of Eyes-Free Touchscreen Keyboard Typing. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility* (Bellevue, Washington) *(ASSETS '13)*. Association for Computing Machinery, New York, NY, USA, Article 69, 2 pages. https://doi.org/10.1145/2513383.2513399

[21] Zheer Xu, Pui Chung Wong, Jun Gong, Te-Yen Wu, Aditya Shekhar Nittala, Xiaojun Bi, Jürgen Steimle, Hongbo Fu, Kening Zhu, and Xing-Dong Yang. 2019. TipText: Eyes-Free Text Entry on a Fingertip Keyboard. 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, 883–899. https://doi.org/10.1145/3332165.3347865

[22] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. arXiv 2205.01068.