EchoTree: Engaged Conversation when Capabilities are Limited

ABSTRACT
We describe the collaborative use of word tree visualizations, *EchoTree*s, to facilitate face-to-face, and remote communication with speech and movement impaired individuals. EchoTree is designed to bridge the inevitable conversational dead space while the impaired person uses assistive technologies to generate written, or artificially spoken sentences. Visualizations that guess multiple conversational directions in which the impaired person might be headed keep conversation partners engaged. Partners may call out possibilities, which the impaired person can confirm or dismiss. Correct guesses accelerate conversational progress. EchoTrees are browser based and interactive. Multiple parties may view, and interact with the same stream of EchoTrees from desktops, tablets, or smartphones. We describe our implementation, and results of an experiment we undertook to identify optimal types of data sources to use for generating the underlying data model.

Author Keywords
assistive technology; prolonged engagement; collaborative conversation; story telling, game

ACM Classification Keywords
H.5.2 Information Interfaces And Presentation: User Interfaces - Interaction styles

General Terms
Human Factors; Design

INTRODUCTION
We collaborate with a motion and speech impaired individual, Henry, who enjoys conversing. Henry's speech impairment is complete. Quadriplegia, while severely limiting, does allow Henry to move his head in affirmation or negation. His hand can operate a mouse button. One of his communication modes is via a text-to-speech system (*tts*). A camera mounted on top of his laptop tracks a confetti sized white dot pasted on the lower left of his glasses. The resulting cursor control allows Henry to hunt down the keys of an onscreen keyboard. On a very good day the resulting speed is 15 words per minute. The tts produces sound once a sentence is complete. Uttering words as Henry types them would work, but this approach makes it difficult for listeners to track the very slowly evolving sentences in their minds. Poor tts performance for some words would additionally impede comprehension, which is supported by context when a full sentence is pronounced in a flow.

Submitted to CSCW'14.
Do not cite, do not circulate.

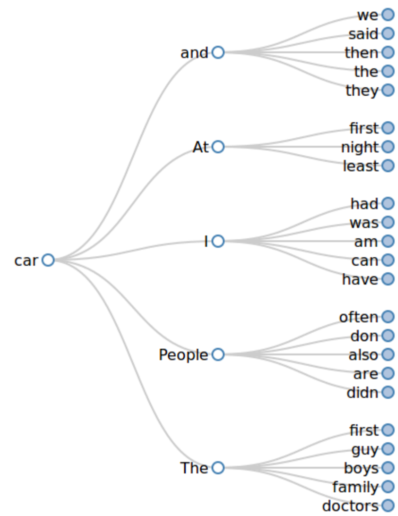

Figure 1. Screenshot of an EchoTree.

This slow communication channel results in very frustrating experiences during gatherings like parties. A guest will make a remark to Henry, who will go to work on an answer. To the conversation partner Henry looks frozen, peering at his laptop screen whose back surface reveals nothing to the expectant partner. Often the potential conversation partner wanders off bewildered before Henry can finish his response sentence.

One improvement would be to install a second display on the backside of Henry's laptop. Inexpensive, USB based options are available. This option would at least allow listener(s) to understand that information is forthcoming. But this solution still keeps the listener(s) passive, and likely to be bored.

In an attempt to ameliorate the situation further we developed EchoTree. EchoTree is a distributed, collaborative word tree. Figure 1 shows an example. As the disabled typist generates written words on their device, such as a laptop, an underlying language model generates word tree visualizations that show multiple conversation threads that are 'likely' to follow each typed word. The tree changes with each completed word. Both the typist and co-present, or telephone listener(s) have access to these trees via any Web browsing devices. Everyone can interact with the trees, exploring them on their device.

Listeners can speak up and suggest possible sentence completions to the typist, possibly saving the typist time, and speeding the conversation. Attempts at predicting content is thus crowd sourced at a small scale. While such attempts to complete sentences for a conversation partner can be annoying for individuals with fully functional communication means, for Henry, the procedure is appreciated, and his family routinely engages in it.

In this writing we present the EchoTree system for the first time. It will become clear in the following sections that many aspects of the design still need to be tested, and the design space must be explored. A user study will, for example, be required. Before such a study, however, a number of system questions must be resolved, so that subsequent user level explorations operate under optimal conditions.

After describing the current user experience and implementation in more detail, we present an experimental EchoTree system level study. The experiment explores which type of data source to use for generating the underlying language model. For questions about prior art, please see our Related Work Section near the end.

USER EXPERIENCE

A word tree[18] is read from left to right, beginning with a single word, the *root*. Branching out from the root are words that might follow the root in an underlying document collection. EchoTree provides five out-branches, which are sorted top to bottom by their likelihood of being the follower word to the root. Each follower word candidate itself features five possible followers to the candidate. The tree thereby presents a number of possible conversation threads that might begin with the root word.

The underlying collection, of course, impacts the follower relationship probabilities. We analyze three types of collections in the experiment section.

EchoTrees are browser applications that can be viewed anywhere, by multiple users. In particular, every word a typist enters on their laptop induces an EchoTree, which is made available by an EchoTree server as an interactive Web application. The typist can see the trees as well. If the word that the typist has in mind to type next is contained in the tree, they can click on the word. The word is transferred to the *sentence box* near the display bottom, which collects the evolving sentence, and saving on cumbersome text input. Once the typist finishes the next word, the previous EchoTree is replaced with a new one, rooted at the latest word. Figure 2 shows highlighted three words that were transferred to an input box by clicking on them, one after the other. Beyond providing a word source for a disabled typist to choose from, this arrangement also enables a number of scenarios where the typist communicates with conversation partners who have access to the trees. The important point about each of these scenarios is that conversation partners remain engaged in the conversation.

For example, the secondary display facing a conversation partner in the option discussed earlier could always show the typists's current EchoTree. Alternatively, a conversation partner's smartphone or tablet can *tune in* to a typist's EchoTrees. In either case, informed by the EchoTree, the partner can guess future words out loud, again saving on typing and enlivening the conversation.

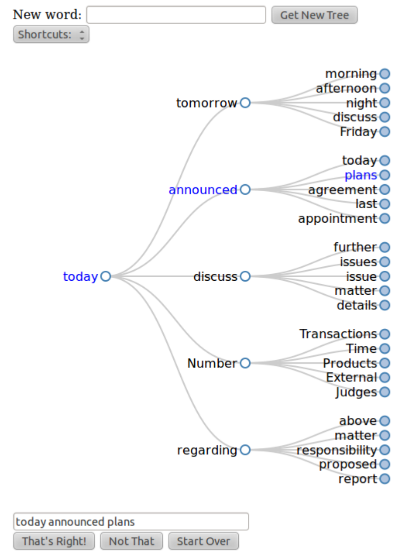

Figure 2. EchoTree re-rooted in ‘today’. A click selected the blue words adding them to the sentence box.

Or, partners can at least spend time exploring EchoTrees on their own while the typist works: The server allows multiple participants to generate their own, separate trees, visible only on their own displays, while still seeing the typist's trees as they are published. The typist's trees are pushed to the client browsers when a new tree becomes available, because the typist finished a word. The pushed trees replace any tree currently displayed on the participant's device.

EchoTree delivery to portable devices is thus subscription based. Participants can in fact subscribe to the EchoTrees of one or more other participants. The primary scenario for this paper is, however, to have conversation participants subscribed just to a disabled person.

The typist or a participant may click or tap on one of the circles in the tree that currently occupies their display. In response, the tree is *re-rooted*: the selected word becomes the root of a new tree. All follow words are recomputed, and a new tree is displayed on the participant's browser, or any other browsers that are subscribed to trees by the person who generated the new tree.

Alternatively, one may type a new word into the text box at the top, and click on the button *Get New Tree*. This action again creates a new tree, rooted at the new word, and displayed everywhere.

The EchoTree facility can be used for a number of purposes. In the context of a disabled typist interacting in a conversation, the facility is used as follows.

Collaborative Conversing

As the typist enters words, and corresponding EchoTrees in the browsers of all tuned in listeners evolve, any word that the typist completes is additionally appended to the sentence box of all participants' screens. As listeners actively think ahead, guess where the typist might be headed, and call out a correct option, the typist can click on the *That's Right* button, or nod.

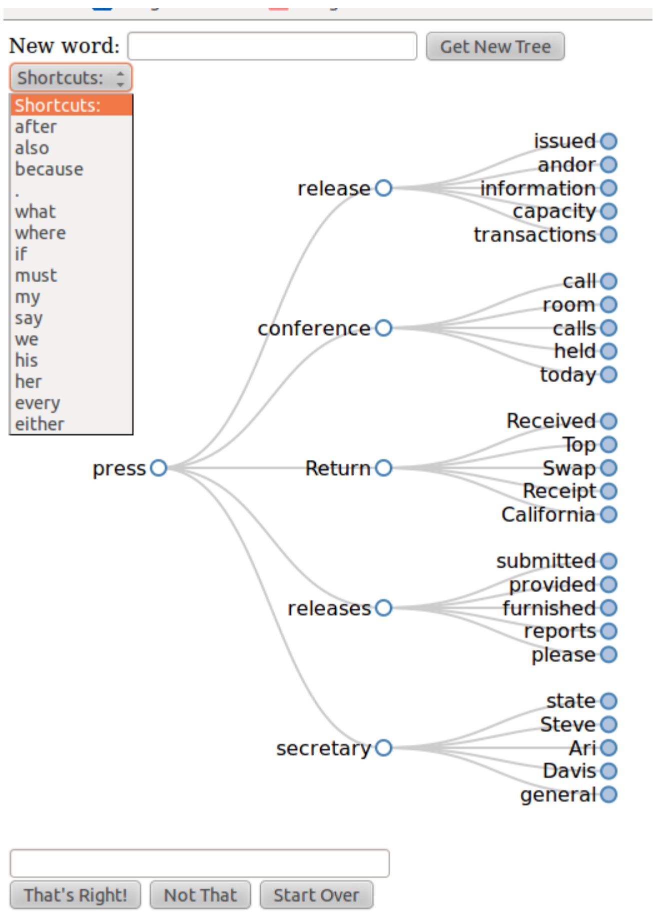

Figure 3. Shortcut words are available as fillers for the sentence box.

After the successful guess the typist continues, skipping one of more words.

Sometimes participants or the typist may wish to enrich sentences with fill words. The pull-down menu below the *New word* field satisfies that need (Figure 3). Selecting any of these words will enter them in the sentence box. Again, in the current implementation this addition appears in all subscribing views of the. Note that the use of EchoTrees for collaborative conversation is not limited to face-to-face situations. Communication with the disabled person via the telephone are also an option. The remote participant tunes into the typist's EchoTrees, and offers guesses over the phone. Since the typist nodding assent is not an option in this scenario, the *Not That* button can serve as a negative response.

Architecture

Figure 4 shows how the EchoTree system is constructed. Central, or distributed EchoTree servers each manage some number of distinct EchoTree channels. All facilities described above operate on one channel. That is all shared EchoTree views are refreshed on that channel, and requests for re-rooting occur on that channel. The server computes trees, given a root word. Once computed, the new tree is pushed to all subscribers.

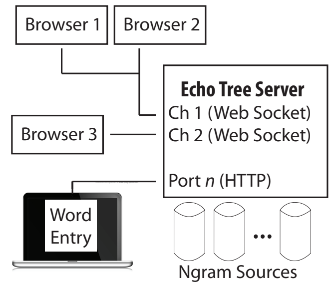

Figure 4. EchoTree architecture. Channels are implemented as WebSocket ports. Browsers tune in to different EchoTree channels.

Multiple, unrelated EchoTree sequences may be served by a single server, using different ports. In Figure 4 Browser three is separated from Browsers one and two, which share mutual EchoTree transmissions.

Browsers communicate with EchoTree servers via WebSocket connections, which are bi-directional. This bidirectionality enables the re-rooting requests from browsers back to the server.

Figure 4 also shows an HTTP port family. These ports are used to push new root words to the echo server from sources other than browsers. Non-standard, desktop based text input applications for disabled typists do not feature Web socket capabilities. The ports can be used instead via standard socket operations. The server listening on those ports interacts with the applications (or bridges to the applications) via the same protocol as the browsers use over Web sockets.

Like new trees created in response to re-rooting requests from browsers, the trees created in response to applications' requests trigger the multicast of a new EchoTree to all browsers on the respective channel. This method allows typists to focus on operating in their usual environment, not being forced to interact with a browser's *New Word* entry to push new root words.

At its bottom, Figure 4 shows a series of databases with word follower frequencies that are the basis for the generation of the trees. The resources in these databases consist of *ngrams* with associated occurrence probabilities. That is, each database holds one or more lists of entries consisting of a probability p, and two or more words, w_0, w_1, $w_2 \cdots$

For example, one snippet of such a list might look like this:

```
1.294000e-05,circumstances,like
1.294000e-05,He,just
1.050000e-04,while,I
1.294000e-05,caused,when
1.294000e-05,not,He
```

The probability informs the EchoTree construction how likely it is that the sequence w_1, $w_2 \cdots$ follows w_0. The probabilities originate from any text collection. But as we will show, the collections, and the choice of ngram arity have important impacts on the generated trees. The EchoTree of Figure 2, for example, is based on bigrams from the Enron email collection [1].

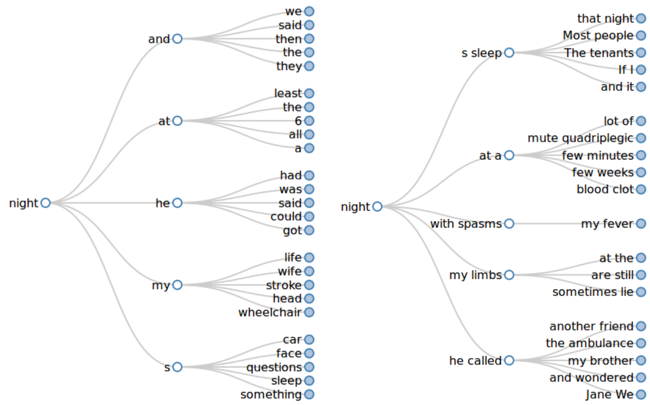

Figure 5. Comparing bigram and trigram tree displays.

Given a new root word, the EchoTree constructor queries one of the underlying databases for the top five list entries whose w_0 is the root word. The choice of which list is consulted is determined by the channel through which the re-root request was delivered, that is on the originating subscription.

Each probability/ngram list is constructed once from one textual source, using the Good-Turing procedure [10]. This treatment smoothes the otherwise spikey ngram distribution, and sets aside some probability mass for ngrams not encountered in the underlying collection.

Cornerstones of EchoTree's system level design are the choices of underlying data sources. But other system level desicions impact the user experience as well. While we address the data source decision in this paper, the following section points to some of these additional considerations.

DESIGN CONSIDERATIONS

We now illustrate how the language model that generates EchoTrees plays into the user experience. As one more example for the details that must be considered, we then look at the issue of stopword removal, to be followed by a section on data source issues that takes us to the associated experiment.

Language Model

The EchoTree word predictions are based on a model of the English language. Working with ngrams has served us well. Both bigrams and trigrams are common choices in Natural Language Processing. Both options produce plausible word trees, but the display is by necessity more cluttered when trigrams are shown. Figure 5 shows a comparison between the two options, using the same root word, *night*. The trigram tree is more expressive than the bigram tree, but it is visually denser. To a disabled typist, finding a word in the trigram display with the intention of then transfering it to the sentence box is more time consuming than scanning the bigram version. Depending on the length of the sought word, the distraction time away from the onscreen keyboard may outweigh the savings in typing.

The distraction time not only comprises the time it takes to visually scan the tree, and to move the cursor to a found word via, for example, a head tracker. But the onscreen keyboard must then be re-acquired for further typing.

Word length, which determines how profitable a cursor excursion to an EchoTree might be, in turn is related to another design choice, the retention or elimination of stop words.

Stopwords

When designing information retrieval facilities, certain very frequently occurring words are often disregarded in user queries, and in underlying index structures. These *stopwords* include words like 'the', 'that', and 'a'. Is stopword elimination appropriate for our purpose of collaborative conversation? We experimented with both options, and included examples for both in the figures. Figure 1 was created while retaining stopwords, while Figure 2 was constructed with stopwords removed.

Clearly, stopword removal tends to display more interesting words. However, the more complete sentence structure borne from stopword retention makes it easier for conversation partners to call out the next word choice, thus saving the disabled typist time, albeit often for short words. Note that this time saving is enjoyed in full, because the typist's attention can continue to dwell on the onscreen keyboard. Short of a formal study, we have anecdotal evidence from Henry, who prefers to lay down all words, not leaving out stopwords in spite of the increased communication expense. A final decision on the stopword question awaits a study.

Data Source

As described, the probability of any word following another is derived from word occurrence statistics in an underlying data source. Which source to use? Intuitively, we might suspect that using a person's own written materials for the ngram statistics will be particularly effective in predicting that person's writing.

Three problems arise from this approach. For one, a disabled person's email messages, for example, will tend to be concise. This brevity in turn limits the amount of material over which ngrams can be computed.

A second problem with using personal writing, like email, is that obtaining the email collection is more difficult now that the material is stored on company servers, than when email was on everyone's personal disk. Not many email users would know how to download their entire email history from their email provider's server. In any event, such a startup effort is unfortunate for any computer application.

Finally, a third drawback of using personal writings is that EchoTrees can be quite revealing. The frequency with which a particular word follows another in one's entire personal corpus should not necessarily be open to public view.

At the other extreme, we can use a broad source, like Google's terabyte ngram collection [14]. Its coverage across millions of Web pages is more or less topic neutral, and thereby maybe universally applicable. Between these extremes lies the option of utilizing multiple ngram sets, each from one topic specific collection of Web pages. When requesting re-rooting, users would choose a configuration that is appropriately close to their topic of conversation. Maybe the topic specificity would be beneficial.

Or, one might argue that neither email, nor Web content, nor ngrams from scanned books [2] are optimal for a conversational context. What if our word choices in conversations is very different from those of our Web pages or email messages?

We conducted an experiment that examined the EchoTree performance of three datasources, each tested under both bigram and trigram design variants.

DATA SOURCE PERFORMANCE EXPERIMENT

Our experiment tested six experimental conditions (independent variables). These conditions arise from tests of three ngram data sources, each producing either bigram, or trigram trees. Our performance measure, the dependent variables comprises two components. Both components were computed automatically.

Performance Measures

The first performance component we computed for each experimental condition is the *reliability* with which EchoTrees predict follow-on words for each successive word in test sentences. The test sentences for each condition were taken from collection set-asides of the condition's data source. No ngrams were collected from those set-asides.

In measuring the reliability of an experimental condition, our automated evaluation pulled one word w after another from each testing sentence S, and constructed an EchoTree with w as its root. If w's follower appeared in the tree at the tree's first level, the sentence was assigned one point. If the follower appeared at the second tree level, one half point was awarded to the sentence. The reliability measure for S was computed as the sum of awarded points, normalized to the length of the sentence. The data source's single reliability performance measure is the average of the individual sentence performances. More formally:

for all sentence S in test sentences **do**
  for all word w_i in S **do**
    tree = computeEchoTree(w_i);
    if w_{i+1} in first level of tree **then**
      $score_s += 1$;
    else if w_{i+1} in second level of tree **then**
      $score_s += 0.5$;
    end if
  end for
  normalize $score$ by length(S);
end for
$reliability = \sum_s score_s / numSentences$;

The second performance measure estimates the *savings* in typing. This measure is the percentage of characters that would not need to be typed in a real life situation, because the respective words were available in EchoTrees. We counted spaces between words in the grand sum of letters to be typed. We also counted the click needed to cause words in the EchoTree to replicate down into the sentence box as a cost equivalent to typing one character. We did not consider the time required for acquiring a word in the tree, and then reacquiring the onscreen keyboard.

Tested Data Sources

Our first data source, consists of 10M Web pages we retrieved from the *Recreation* section of the Open Directory Project (ODP) [3]. The project uses human input to categorized Web pages. We eliminated from the crawl all pages that stemmed from the ODP site itself, retaining pages from the Recreation target sites one level deep. We removed all HTML and Javascript from the resulting pages, then tokenized the remainder using the Stanford NLP Tokenizer [11]. We then extracted 5,055,284 bigrams, and 28,423,891 trigrams.

Our second data source is the Fisher collection's transcripts of 11,000 ten-minute telephone conversations [6, 7]. Each conversation centers around one topic that was assigned to the paid conversation dyad participants. We cleaned the collection of embedded metadata, like the timestamp and source of each speech turn. The remainder was tokenized and we extracted 149,789 bigrams and 93,534 trigrams.

Our third experiment data source, finally is a blog our collaborator maintains [9]. The snapshot was taken on May 10, 2013, and comprises 1,167 sentences (96.5MB), generating 12,222 bigrams, and 16,567 trigrams after a 5% set-aside.

Test Procedure

For the ODP Recreation Web source, we removed HTML markup before further processing. Before extracting ngrams, from data source content, we set aside a portion of the corpus as a source for test sentences. We used a 10% set-aside for the Recreation source and the Fisher conversations, and 5% for the blog source.

As explained earlier, we computed ngram probabilities using the Good-Turing algorithm [10].

The end result were six lists. Each of the three data sources induced one list of bigrams, and another list of trigrams.

We randomly selected 60 sentences from each of the corpus set-asides. For calrity we explain the further procedure using one of the six experimental conditions: bigrams from the Fisher collection of conversations.

The test software tokenized each of the 60 sentences that were set aside from the Fisher collection before generating its bigrams. Using the Fisher bigram list, word trees were computed repeatedly as described in the Performance Measures section. The trees were not displayed; only their internal representations were used for the computations. This procedure produced a *csv* file with one line for each sentence, recording the two performance scores: The savings in typing and the reliability in predicting follow-on words.

RESULTS

We used a oneway ANOVA to evaluate the differences among mean typing savings (first dependent variable), and among reliability measure means (second dependent variable) for the

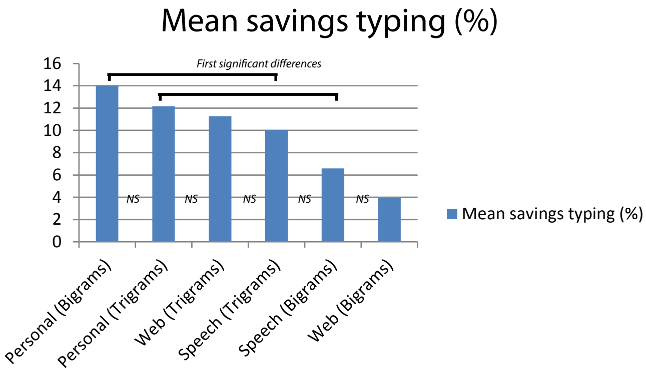

Figure 6. Mean savings in typing. No neighboring differences are significant (*ns*)

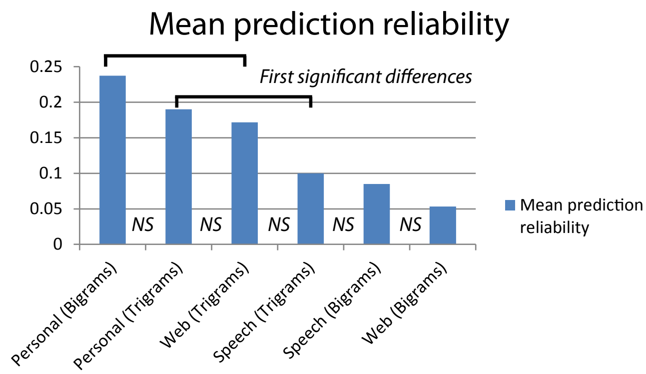

Figure 7. Mean prediction reliability. No neighboring differences are significant (*ns*)

six experimental conditions, i.e. the six combinations of three data sources and two ngram arities.

Levene's test found variances among the six groups to be different. We nevertheless opted for the ANOVA procedure, because our sample sizes were all equal at 60 sentences.

We found a highly significant effect of the data source on savings in typing, $F(5, 354) = 16.0$, $p < .001$. The effect of data source on prediction reliability was also highly significant, $F(5, 354) = 25.4$, $p < .001$.

Tukey HSD post-hoc tests revealed a number of significant between group differences. See Table 1 and Table 2.

Figure 6 shows a comparison of mean savings in keyboarding effort.

Figure 7 displays the corresponding results for prediction reliability.

DISCUSSION

As suspected, results for ngrams from the personal collection were optimal. However, note that all differences between $Personal_{bi}$ and its successively less effective neighbors were not significant until $Speech_{tri}$. Similarly for $Personal_{bi}$ to $Speech_{bi}$ This outcome is bad news for bigrams, but good news for trigrams. If we want to use bigram EchoTrees because, for instance, they are quicker for users to scan, then the only optimal solution is the use of personal collections. The next option is to use $Speech_{bi}$ which lies below $Personal_{bi}$ by about 8 percentage points.

For trigram EchoTrees the result is more favorable. Rather than having to use personal collections, we could use Web_{tri} or $Speech_{tri}$. Given the drawbacks we described of using personal collections, this result argues for the use of trigrams. On the other hand, the possible detriments of clutter that go along with trigrams pulls towards bigrams. Further end user studies are required to make a final decision.

We were surprised by the relative strength of Web_{tri}. (Note that the difference between Web_{tri} and Web_{bi} is significant.). We did not expect the Web Recreation crawl to do well for either variant, because the raw results from a crawl are very 'messy.' Even after HTML tag removal, a large amount of formatting, and styling information is left over. Numerous menu headings, and boilerplate materials are present as well. We made no effort to clean the corpus beyond HTML tag removal.

Just to ensure that the random draw of Web_{tri} test sentences did not produce an unusual outcome, we repeated Web_{tri} with a different set of random 60 sentences from the respective set-aside. The result was comparable to Figure 6.

The prediction reliability (Figure 7) shows that the order of the six conditions by decreasing performance is as per the savings measure. A fundamental difference between the two measures, however, is that reliability does not depend on individual word lengths.

The best performer in reliability was $Personal_{bi}$ with just under .25 on the scale from 0 to 1. Again, for bigram trees no choice is better than $Personal_{bi}$. The next bigram choice is $Speech_{bi}$, which is a significantly worse performer for prediction. For trigrams, however, the equivalent choice of Web_{tri} is available.

In summary, the optimal choice for ngram data source is always the personal collection. This result holds for both bigram and trigram trees.

For bigrams, the next best choice is to use the Fisher conversation transcripts, albeit at large cost in typing savings and prediction reliability.

If trigrams are acceptable to end users, then the recreation Web crawl is an acceptable choice for both savings and reliability.

FUTURE WORK

Clearly, end user experiments are required to resolve at least two questions. First, are bigrams, or trigrams the preferable tree form for the purpose of supporting conversations with the disabled typist? Second, do conversation partners generate correct guesses that are different from the literal content of the EchoTree they scan, but still capture the typist's intention?

If such associative guesses do occur, then the overall system—machine plus human(s)—will exceed the performance measures of our experiment. Our results show a lower

	$Personal_{bi}$ M=14.0 SD=9.8	$Personal_{tri}$ M=12.1 SD=8.4	$Speech_{bi}$ M=6.6 SD=7.5	$Speech_{tri}$ M=10.1 SD=8.0	Web_{bi} M=4.0 SD=3.5	Web_{tri} M=11.3 SD=5.3
$Personal_{bi}$;M=14.0,SD=9.8			p<.001	p<.045	p<.001	
$Personal_{tri}$;M=12.1,SD=8.4			p<.05		p<.001	
$Speech_{bi}$;M=6.6,SD=7.5	p<.001	p<.05				p<.05
$Speech_{tri}$;M=10.1,SD=8.0	p<.05				p<.001	
Web_{bi};M=4.0,SD=3.5	p<.001	p<.001		p<.001		p<.001
Web_{tri};M=11.3,SD=5.3			p<.05		p<.001	

Table 1. Post-hoc test results for savings in typing.

	$Personal_{bi}$ M=.237 SD=.145	$Personal_{tri}$ M=.190 SD=.119	$Speech_{bi}$ M=.085 SD=.143	$Speech_{tri}$ M=.010 SD=.091	Web_{bi} M=.053 SD=.052	Web_{tri} M=.172 SD=.082
personal bi M=.237,SD=.145			p<.001	p<.001	p<.001	p<.05
personal tri M=.190,SD=.119			p<.001	p<.001	p<.001	
speech bi M=.085,SD=.143	p<.001	p<.001				p<.001
speech tri M=.010,SD=.091	p<.001	p<.001				p<.05
web bi M=.053,SD=.052	p<.001	p<.001				p<.001
web tri M=.172,SD=.082	p<.05		p<.001	p<.05	p<.001	

Table 2. Post-hoc test results for prediction reliability.

bound.The result for the ngram arity will impact the range of available data source choices.

A number of follow-on experiments will help illuminate the EchoTree design space further. For example, note a difference between the recreation Web crawl and the Fisher conversations source. The former is biased towards recreation, while the conversations range over 40 topic areas. The Fisher test sentences were drawn from all those topics. The relatively high performance of Web_{tri} suggests that multiple sources should be used for EchoTrees, and the proper one chosen based on the topic of the typing activity. This approach would be more complex, but feasible. For this reason the EchoTree architecture (Figure 4) anticipates multiple ngram sources that can be switched dynamically.

Another option is a learning component, which would over time adjust ngram weights to each user. Both, computational linguistics, and machine learning algorithms can help in this regard. This approach would be a hybrid between personal and stock data sources.

A number of optimizations are also in our plan. For example, word stemming is a likely improvement on prediction reliability. Related techniques are well known [16]. We also plan to broadcast each character that the typist enters, rather than just the entire word once it is completed. This letter by letter dissemination will allow conversation partners to provide not just follower word prediction, but also word completion.

RELATED WORK

EchoTree serves as an example of *Augmentative and Alternatie Communication* (AAC), defined broadly as 'attempts to study and when necessary compensate for temporary or permanent impairments, activity limitations, and participation restrictions of individuals with severe disorders of speech-language production and/or comprehension including spoken and written modes of communication.' [5]. Over the past few decades, AAC technologies have closely tracked developments in communication technologies and adapted them to improve the rate, fluidity, and efficacy of communication for users with physical or speech impairments [17].

Advances in language modeling and prediction have been fruitful for AAC researchers; anticipating imminent letters, words, or even sentences can vastly reduce text input demands of impaired users. Systems such as Humsher [12], for example, have adapted letter-based text-entry systems developed for able-bodied users to significantly accelerate text entry for severely motor-impaired users. While navigating systems such as EchoTree which rely on n-gram based prediction can impose additional cognitive overhead, prior work has found that this cost is outweighted by the benefit of increased typing speed [15].

EchoTree builds not only on prior research in n-gram word prediction, but also on prior work in technology-assisted co-construction of sentences. Prior efforts have shown how involving an able-bodied communication partner in the conversation loop to make guesses about the intended messages can complement the benefits offered by ngram language models [13].

Similarly, the visual design of EchoTree adapts the interactive "keyword-in-context" visualization technique embodied in the Word tree [18] from its original purpose of retrospective corpus studies to the new problem of collaborative text generation. The use of visual metaphors, direct interaction, and output-as-input techniques allow for tight coupling between conversation participants and the information display [4]. EchoTree's sentence box and confirmation loop further afford the process of grounding, enabling tighter coupling between the conversation partners themselves as they negotiate the evolving sentence together. [8].

CONCLUSION

We introduced EchoTrees, which are distributed, interactive word trees. The facility is designed to engage conversation partners who interact with motion and speech disabled individuals. The word trees, which are multicast over the Web are browser applications that allow the partners to see what the typist has written so far. In addition, after each word is typed, a tree of possibly following words is recursively constructed and distributed. The disabled person can use the EchoTrees as a source for words that then don't need to be typed. The conversation partners can use them to guess future words, thus again saving the typist time and effort.

We explained some of the design problems and considerations, and then provided an empirical analysis of which data sources are best to use for constructing language models to use in this interactive conversation scenario.

Final word is out on the question whether users will prefer bigrams or trigrams in their EchoTree experience. We listed the pros and cons for both choices, and their impact on the system level decisions. Studies of user level behaviors and preferences will need to be conducted as well; we listed some of the topics in question.

The problem of social isolation for communication impaired individuals is tough to solve for the general case. But special, computer supported solutions can provide impetus towards addressing this isolation. Computers are patient, conversation partners are not. EchoTrees try to bridge that gap by crowd sourcing some of the conversational moves.

REFERENCES

1. The Enron email dataset. World-Wide Web, August 2009. Archived by WebCite® at http://www.webcitation.org/6AniNSXC6. Accessed: 2012-09-19.
2. The American National Corpus. World-Wide Web, 2012. Archived by WebCite® at http://www.webcitation.org/6AnivCgHX. Accessed: 2012-09-19.
3. The Open Directory Project. http://www.dmoz.org/, April 2013.
4. Ahlberg, C., and Shneiderman, B. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. Human Factors in Computing Systems (CHI 1994), ACM (Boston, MA, USA, April 1994), 313–317.
5. (ASHA), A. S.-L.-H. A. Roles and responsibilities of speech-language pathologists with respect to augmentative and alternative communication: position statement. Position Statement, 2005.
6. Cieri, C., Graff, D., Kimball, O., Miller, D., and Walker, K. Fisher english training speech part 1 transcripts. http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2004T19, December 2004.
7. Cieri, C., Graff, D., Kimball, O., Miller, D., and Walker, K. Fisher english training part 2, transcripts. http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2005T19, April 2005.
8. Clark, H. H., and Brennan, S. A. *Perspectives on socially shared cognition*. No. 7. APA Books, 1991, ch. Grounding in communication.
9. Evans, H. Hevans. http://hevans-hevans.blogspot.com/, May 2013.
10. Gale, W. A. Good-turing smoothing without tears. *Journal of Quantitative Linguistics 2* (1995).
11. Manning, C., Grow, T., Grenager, T., Finkel, J., and Bauer, J. Stanford tokenizer. http://nlp.stanford.edu/software/, 2013.
12. Poláček, O., Míkovec, Z., Sporka, A. J., and Slavík, P. Humsher: A predictive keyboard operated by humming. ASSETS '11, ACM (Dundee, Scotland, UK, October 2011), 75–82.
13. Roark, B., Folwer, A., Sproat, R., Gibbons, C., and Fried-Oken, M. Towards technology-assisted co-construction with communication partners. 2nd Workshop on Speech and Language Processing for Assistive Technologies (SLPAT '11), ACL (Edinburgh, Scotland, UK, July 2011), 22–31.
14. Thorsten Brants, A. F. Web 1t 5-gram version 1. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13, September 2006.
15. Trnka, K., McCaw, J., Yarrington, D., McCoy, K. F., and Pennington, C. User interaction with word prediction: The effects of prediction quality. *ACM Transactions on Accessible Computing 1*, 3 (February 2009), Article 17.
16. van Rijsbergen, C., Robertson, S., and Porter, M. New models in probabilistic information retrieval.
17. Vanderheiden, G. C. A journey through early augmentative communication and computer access. *Journal of Rehabilitation Research & Development 39*, 6 (Nov/Dec 2002), 39–53.
18. Wattenberg, M., and Viégas, F. B. The word tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov/Dec 2008), 1221–1228.