Alison Smith, Varun Kumar, **Jordan Boyd-Graber**, Kevin Seppi, and Leah Findlater. **User-Centered Design and Evaluation of a Human-in-the-Loop Topic Modeling System**. *Intelligent User Interfaces*, 2018, 12 pages.


```
@inproceedings{Smith:Kumar:Boyd-Graber:Seppi:Findlater-2018,
Author = {Alison Smith and Varun Kumar and Jordan Boyd-Graber and Kevin Seppi and Leah Findlater},
Booktitle = {Intelligent User Interfaces},
Year = {2018},
Title = {User-Centered Design and Evaluation of a Human-in-the-Loop Topic Modeling System},
Url = {http://cs.umd.edu/~jbg//docs/2018_iui_itm.pdf},
}
```


Alison won a best student paper honorable mention (3 out of 300)

Contact Jordan Boyd-Graber (jbg@boydgraber.org) for questions about this paper.

Closing the Loop: User-Centered Design and Evaluation of a Human-in-the-Loop Topic Modeling System

Alison Smith
University of Maryland
Computer Science
College Park, Maryland, USA
amsmit@umd.edu

Varun Kumar
University of Maryland
Computer Science
College Park, Maryland, USA
varunk@cs.umd.edu

Jordan Boyd-Graber
University of Maryland
CS, iSchool, UMIACS, LSC
College Park, Maryland, USA
jbg@umiacs.umd.edu

Kevin Seppi
Brigham Young University
Computer Science
Provo, Utah, USA
kseppi@byu.edu

Leah Findlater
University of Washington
Human Centered Design and Engineering
Seattle, Washington, USA
leahkf@uw.edu

ABSTRACT

Human-in-the-loop topic modeling allows users to guide the creation of topic models and to improve model quality without having to be experts in topic modeling algorithms. Prior work in this area has focused either on algorithmic implementation without understanding how users actually wish to improve the model or on user needs but without the context of a fully interactive system. To address this disconnect, we implemented a set of model refinements requested by users in prior work and conducted a study with twelve non-expert participants to examine how end users are affected by issues that arise with a fully interactive, user-centered system. As these issues mirror those identified in interactive machine learning more broadly, such as unpredictability, latency, and trust, we also examined interactive machine learning challenges with non-expert end users through the lens of human-in-the-loop topic modeling. We found that although users experience unpredictability, their reactions vary from positive to negative, and, surprisingly, we did not find any cases of distrust, but instead noted instances where users perhaps trusted the system too much or had too little confidence in themselves.

INTRODUCTION

Topic modeling helps users understand vast document collections when there are too many documents or too little time to read individual documents: a journalist processing the reports surrounding breaking news or a legal team finding interesting e-mails during discovery. Topic modeling automatically discovers the themes (topics) in large corpora of unstructured text [7] by modeling the intuition that groups of words that form a common theme appear more often together than not; the discovered topics are sets of words (e.g., "football, touchdown, NFL, . . . "), and each document can contain multiple topics. However, traditional topic models can include poor quality topics [30] or can be misaligned with a domain expert's understanding of the corpus [23]. Pre-processing techniques and parameter tuning [36] can improve model quality, but typically require many time-consuming iterations and can only be performed by an algorithm expert. Human-in-the-loop topic modeling (HL-TM) aims to solve these problems by allowing end users who are not experts with topic modeling algorithms to directly refine the model (e.g., changing which words are included in a topic, or merging or splitting topics).

HL-TM has largely implemented topic model refinements based on what algorithm developers *assume* users want, as well as algorithm concerns, such as which refinements are the most straightforward to implement [23, 10, 21, 26]. In contrast, Lee et al. [27] employed a user-centered approach to identify a set of topic refinement operations that users expect to have in a HL-TM system. However, because no existing implementation supported the set of refinement operations (e.g., add a topic word, change word order, merge topics), they used Wizard-of-Oz refinements: the resulting topics were updated superficially—not as the output of a data-driven statistical model (the goal of topic models). Thus, this study is limited, as it ignores the common interactive machine learning issues that arise in a fully interactive system—such as, unpredictability, latency, and complexity—that can affect user experience and impact how users interact with a HL-TM system.

To address these limitations we implemented a broad set of refinements operations [27, 31] in a single system: *add word*, *remove word*, *remove document*, *change word order*, *split topic*, *merge topics*, and *add to stop words*. Unlike in Lee et al. [27], where refinements were immediately applied only in the interface and were not saved to the backing model, in our study

IUI 2018, March 7–11, 2018, Tokyo, Japan.

ACM ISBN 978-1-4503-4945-1/18/03 ...$15.00.
https://doi.org/10.1145/3172944.3172965

participants chose when to "save" and update the backing model with their refinements. We evaluated this fully interactive, user-centered HL-TM system with 12 participants who were not experts in topic modeling. This system is also fast enough to support (and study) interactive use of fine-grained topic modifications, unlike prior systems [21, 10, 23] (e.g., early HL-TM systems took $5-50$ seconds for each update [23], whereas ours took $.09-.63$ seconds to update after saving during the user study). In addition to observing how participants employed the refinements to improve a topic model, we qualitatively assessed overall experience and the potential impacts of common interactive machine learning challenges, such as unpredictability and latency.

Participants subjectively and objectively improved topic model quality using the given refinements. Similar to Lee et al. [27], simple refinements, such as *remove word* and *change word order* were most common, however, unlike Lee et al. [27], usage did not align perfectly with perceived utility, as participants found *change word order* to be one of the least useful refinements. Our fully interactive system also exposed previously hidden issues: unexpected results from using refinements and an inability to track model changes. Participants also varied in when and why they saved their local updates to the underlying model. Additionally, participants who had more trust in the system (or perhaps less confidence in themselves) were not as frustrated by unpredictability as others.

This work makes the following contributions: (1) efficient asymmetric prior-based implementation of a broader set of user-centered refinement operations than has been previously implemented in a single system; (2) extension of a previous, limited HL-TM user study [27] to this fully interactive system; (3) understanding of how interactive machine learning issues such as unpredictability, trust, and lack of control impact HL-TM usage and design; and (4) design principles for user-centered HL-TM systems.

BACKGROUND

Topic modeling automatically identifies the themes or topics that occur in a collection of documents. A common approach is Latent Dirichlet Allocation (LDA) [4], which is a generative statistical model that models each document as a distribution of topics and each topic as a distribution of words. Here, we provide a brief background of LDA and existing HL-TM systems. We also discuss design challenges for interactive machine learning systems more generally.

Latent Dirichlet Allocation

LDA [4] is a generative model, which assumes that each document d is generated from a fixed set of k topics. Each topic is a multinomial distribution, ϕ_z, over the vocabulary, V. Each instance of a word, or token, w_i, is generated by sampling a topic assignment z_i from the document's topic distribution θ_d, followed by sampling the selected topic's distribution ϕ_{z_i}, to generate a token w_i.

The multinomial distributions θ_d and ϕ_z are drawn from Dirichlet distributions that encode sparsity—how many words you expect to see in a topic or how many topics in a document—and can also incorporate expert knowledge from users. Below, we show how to adjust the Dirichlet hyperparameters, α and β, to encode user information about documents (α) and topics (β).

Given this model, we need to find the distribution that best explains our observed documents. Griffiths and Steyvers [17] propose a collapsed Gibbs sampling based method. In collapsed Gibbs sampling, the probability of a topic assignment $z = t$ in document d given an observed token, w_i, and the other topic assignments, $z_{\_}$, is

$$P(z = t \,|\, z_{\_}, w) \propto (n_{d,t} + \alpha)\frac{n_{w,t} + \beta}{n_t + V\beta}. \quad (1)$$

Here, $z_{\_}$ are the topic assignments of all other tokens, $n_{d,t}$ is the number of times topic t is used in document d, $n_{w,t}$ is the number of times token w is used in topic t, and n_t is the marginal count of the number of tokens assigned to topic t. For traditional topic models, the Gibbs sampler assigns latent topics Z for all tokens in the corpus, going over all the documents in the corpus repeatedly until the algorithm converges.

The state of the sampler represents the algorithm's best guess of the topic assignments for every token. Adding a human in the loop requires the user to be able to inject their knowledge and expertise into the sampling equation to guide the algorithm to better topics.

Human-in-the-loop Topic Modeling

Compared to traditional tools that visualize static topic models [15, 13, 9], HL-TM provides mechanisms to allow end users to refine *changing* topic models. Numerous tools have been designed around this concept, each implementing a variety of refinements, but without extensive user studies on real world-tasks to show the refinement implementations match user expectations. Andrzejewski et al. [3], building on Boyd-Graber et al. [6], introduced a statistical framework for users to specify pairs of words that should or should not belong to the same topic through "must-link" and "cannot-link" constraints. This framework has been extended by numerous HL-TM tools [23, 11, 33]; however, this approach limits the possible set of refinements that can be supported and does not match users expectations in that end users typically do not think of specifying model refinements as pairwise word correlations [27].

Other HL-TM tools take alternative approaches, such as UTOPIAN [10], which uses nonnegative matrix factorization and supports creating, splitting, and merging topics, in addition to changing word weights. Lund et al. [28] also uses a fast matrix-factorization approach but only supports a limited set of interactions. ConVisIT [21] uses fragment quotation graphs and supports splitting and merging topics. Finally, Distillery [31] uses informative priors to support merging and removing topics, as well as adding and removing topic words and adding to stop words. However, no prior implementation supports the complete set of refinement operations requested by users [27].

Interactive Machine Learning Design Challenges

Interactive machine learning—also known as mixed-initiative or human-in-the-loop systems—incorporate human input to produce an output or a decision. Designers of these must account for challenges inherent in machine learning, which deviates from traditional user interface design. While user interfaces should provide immediate updates [29], be predictable [18], ensure the user feels in control, and reduce short term memory load [34], interactive machine learning systems are commonly slow, unpredictable, share control between user and system, and are complex [2, 19].

Prior work has examined how predictability, control, and accuracy affect the user experience. Gajos et al. [14] showed that increasing predictability and accuracy lead to improved satisfaction, while Kangasraasio et al. [24] showed that allowing users to see the predicted effects of an action before committing to it can improve task performance and acceptance. Similarly, users of PeerFinder—a tool that recommends similar students based on academic profiles—were more confident and engaged when given more control even with the negative effect of added complexity [12]. In this work, we explore how users are affected by complexity, unpredictability and lack of control in HL-TM.

REFINEMENT IMPLEMENTATION

Prior work [27, 31] identified a set of refinements that users expected to be able to use in a HL-TM system. There is no implementation for the broad set of user preferred refinements, so Lee et al. [27] simulated refinements using a Wizard-of-Oz method. To truly evaluate user experience with a fully functional HL-TM system, we implemented seven refinements requested by users: *add word*, *remove word*, *change word order*, *remove document*, *split topic*, *merge topic*, and *add to stop words*.

These refinements include the six top refinements identified, but not implemented, by Lee et al. [27], except for *merge words*. *Merge words* was discussed in that study as a means for organizing topic words in the interface rather than a deeper specification that should be implemented in the model. We also included two refinements that were not suggested by users in Lee et al. [27], perhaps due to that study's method: *merge topics* did not arise because users only refined individual topics and *add to stop words* may have been overlooked because the study used a generic corpus with a well-curated stop words list.

Refinement Implementation

When a user provides feedback to a topic model, we view this as correcting an error the model made. We can thus divide this feedback into two broad classes: *forgetting* bad things the model learned and *injecting* new knowledge into the model. Forgetting is accomplished by invalidating the topic-word assignments for targeted word types. This is equivalent to the model seeing that word for the very first time, allowing it to make better decisions. In tandem with forgetting, injecting provides hints that encourage the algorithm to make better decisions going forward.

Injecting information happens through modifying the Dirichlet parameters for each document, α, and each topic, β.[1] Recall the Gibbs sampling conditional probability $P(z = t \mid z_{\_}, w)$ (1), which has two parts: how much a document likes a topic—$(n_{d,t} + \alpha_{d,t})$—and how much a topic likes a word—$(n_{w,t} + \beta_{w,t})$. The priors are added to the topic assignment counts; thanks to the conjugacy of multinomial and Dirichlet distributions, these priors are sometimes called "pseudo-counts". Our HL-TM takes advantage of this by creating pseudocounts to encourage the changes users want to see in the topic.

The refinement operations are:

1. *Add word*: to add the word w to topic t we forget w from all other topics and encourage the Gibbs sampler to assign topic t for all of the word's tokens, w_i. For the former, we forget the tokens' topic assignments. For the latter, we increase the prior of w in t by the difference between the topic-word counts of w and topic's top word w' in topic t (i.e., $n_{w',t} - n_{w,t}$).
2. *Remove word*: to remove the word w from topic t we need to forget all the word's tokens w_i from t and discourage the Gibbs sampler from reassigning t to the word w. To discourage the sampler from assigning w to t with a high probability, we assign a very small prior,[2] ε, to w in t.
3. *Change word order*: to reorder word w_2 to appear before word w_1 in topic t we need to ensure that w_2 is ranked higher than w_1 in the topic t. To enforce this, we increase the prior of w_2 in t by the difference between the topic-word counts (i.e. $n_{w_1,t} - n_{w_2,t}$). Intuitively, this operation resembles providing supplemental counts to w_2 so that it ranks higher than w_1 in the topic.
4. *Remove document*: in LDA, each document can be represented as a probability distribution over topics. To remove the document d from topic t, we forget the topic assignment for all words in the document d and assign a very small prior,[2] ε, to the topic t in α_d.
5. *Merge topic*: merging topics t_1 and t_2 means the model will have a combined topic that represents both t_1 and t_2. We assign t_1 to all tokens that were previously assigned to t_2. This effectively deletes t_2 from the model and decrements the number of topics.
6. *Split topic*: to split topic t the user provides a subset of the topic's words, or seed words, which need to be moved from the original topic, t, to a new topic, t_n. To implement this, we invalidate the original topic assignment of all seed words, create a new topic by incrementing the number of topics, and assign a large prior for each of the seed words, w_s, in the new topic, t_n. The Gibbs sampler's job is to sort which words land in which of the new child topics.
7. *Add to stop words*: adding the word w to global stop words removes w from *all* topics. We exclude that w from the vocabulary, V. This ensures that the Gibbs sampler will ignore all occurrences of w in the corpus.

[1] To implement these refinement operations, we make use of the vector interpretation (rather than scalar) of these priors. Thus, α_d is a K dimensional vector for each document d and ϕ_k is a V dimensional vector for each topic k.

[2] We use $\varepsilon = 0.000001$ for our experiments.

Interface

The HL-TM user interface (Figure 1) represents a topic model as a list of topics on the left panel, each displayed as their first three words. Selecting any topic in the list shows the full topic view in the right panel, which consists of the top 20 topic words and snippets of the top 40 topic documents. Documents are ordered by their probability for the topic t given the document d, or $p(t|d)$. Each word, w, is ordered and sized by its probability for the topic t, or $p(w|t)$; this simple word list representation provides users a quick topic understanding [1, 35]. Hovering or clicking on topic words highlights the word in the displayed document snippets.

Users refine the topic model using simple interactive mechanisms. We require users to click "save" to incorporate their specified refinements instead of applying them immediately because the system does not support reverting the model after an update (we discuss batch vs. immediate refinements in Discussion and Future Work). Instead, the interface displays intermediate feedback, such as bold and italicized words, representing users' specified refinements before saving, and any or all of the outstanding refinements can be undone. When users press "save", their specifications are incorporated into the model (Refinement Implementation).

QUALITATIVE EVALUATION OF A HL-TM SYSTEM

Our fully interactive user-centered HL-TM system focuses on topic model novices. Participants explored and refined a model built from a Twitter corpus of complaints about airlines, followed by a semi-structured interview. The study focused on a broad set of operations in a fully interactive system (compared to [27]), as well as understanding how interactive machine learning challenges–predictability, complexity, and latency–complicate topic modeling. For refinements in HL-TM, we divide predictability into *control* and *stability*, where *control* is how much the user's refinement is reflected after the model updates (e.g., a specified word is added to the topic), and *stability* is how many other changes not specified by the user appear in the model (e.g., other unspecified words are added). Instability, in particular, is a concern with HL-TM: small changes to the model can propagate in unexpected ways.

Method

The study protocol included a training task to familiarize users with topic modeling, a test task to refine a topic model, and a semi-structured interview on the experience.

Participants

We recruited twelve participants (five male, seven female) from campus e-mail lists. They were on average 30.5 years old ($SD = 10.3$) and fluent English speakers. Educational backgrounds included human-computer Interaction (5), information management (2), education (1), mechanical engineering (1), computer science (1), psychology (1), and international government (1). Experience with topic modeling varied (nine with no experience, three with limited) as did experience with data science or machine learning (seven with no experience, three limited, two significant). Each participant got a $15 Amazon gift card. We refer to participants as P1–P12.

Dataset and Topic Model

We used a separate dataset and model for the training and test tasks. For *training* we generated a model with 10 topics from a dataset of 2,225 BBC news articles corresponding to stories in five topical areas (business, entertainment, politics, sports, tech) from 2004 – 2005 [16]. For the *test* we used the Twitter US Airline Sentiment dataset from Kaggle,[3] which includes 14,485 total tweets from February 2015 directed to six popular airlines (American, Delta, Southwest, United, US Airways, Virgin America). The dataset includes manually applied labels organizing the tweets into "positive" (2,363 tweets), "neutral" (3,099 tweets), and "negative" (9,178 tweets) sentiment categories. We modeled the 9,178 negative sentiment tweets with 10 topics using a standard stop words list[4] and 300 Gibbs sampling iterations. For each subsequent update during the task, 30 Gibbs sampling iterations were run. Table 1 shows the initial set of topics (henceforth T1–T10). We automatically computed topic quality for each topic using a topic coherence metric based on Normalized Pointwise Mutual Information [5, NPMI] with Wikipedia as the reference corpus [25].

Procedure

Sessions were designed to take one hour, but in practice took up to 90 minutes, and they were conducted remotely with audio and screen-capture recording. We introduced participants to topic modeling and to the HL-TM tool using the *training topic model*. The interviewer described each refinement operation and asked the participant to practice sample operations.

Participants then reviewed the raw tweets of the *test* dataset in a csv file and were told to imagine they had been asked to organize these tweets to identify different classes of airline complaints. They then opened the HL-TM tool with the *test topic model* (Figure 1) and were instructed that an initial model of 10 topics had been generated to help summarize the tweets, but that they may notice flaws and may need to refine the model. The interviewer asked a few introductory questions about the model and the tool, then instructed participants to think aloud while refining the model using the tool until they felt it best categorized the tweets into types of complaints. Participants were given a maximum of 20 minutes for the task, and afterwards they answered semi-structured interview questions about the task, model, and tool.

Data and Analysis

We logged user interaction with the HL-TM tool, including the state of the model at each iteration, when the user pressed "save", and refinement usage. The task audio was also transcribed and coded along with the responses for the post-task interview. Coding followed a thematic analysis approach [8] to uncover the overarching themes represented by more specific codes within the data. The codebook was organized into five themes containing a total of 40 codes: challenges (10 codes), tool requests (10), refinement requests (8), save strategies (6), and refinement strategies (6). To determine agreement, two researchers independently coded transcripts for a random participant. Of 21 instances, the researchers agreed on the codes for 12 and disagreed on nine. Disagreements were resolved

[3] https://www.kaggle.com/crowdflower/twitter-airline-sentiment

[4] https://raw.githubusercontent.com/mimno/Mallet/master/stoplists/en.txt

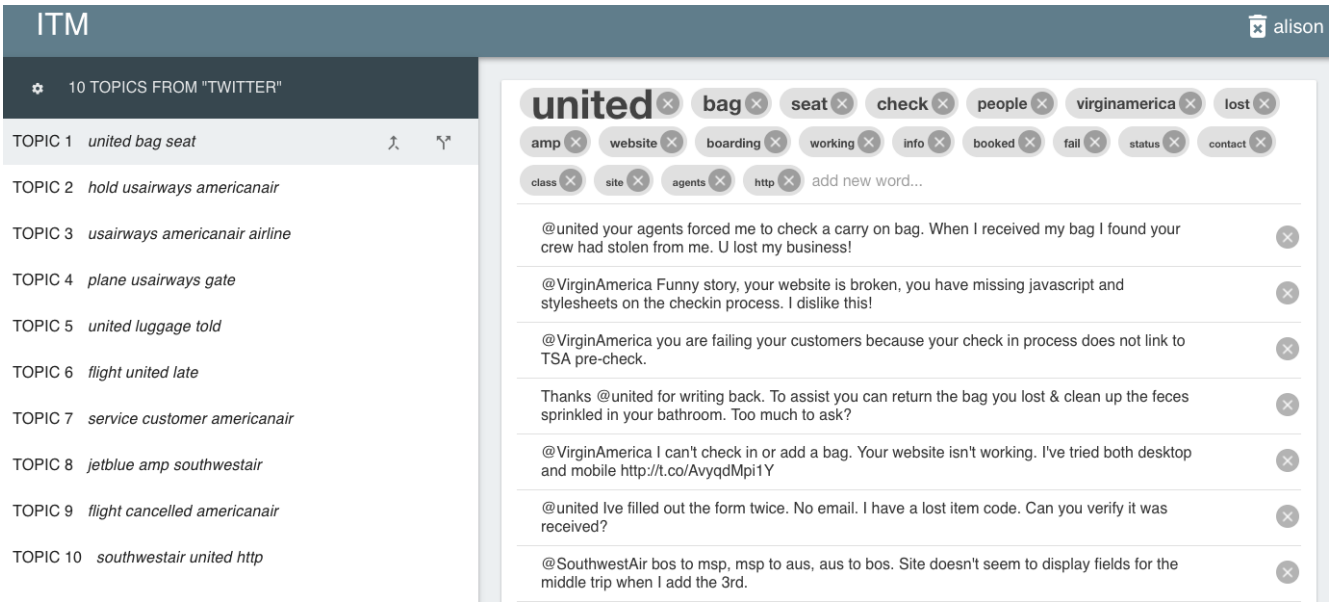

Figure 1. User interface for the HL-TM tool. A list of topics (left) are represented by topics' first three topic words. Selecting a topic reveals more detail (right): the top 20 words and top 40 documents. Hovering or clicking on a word highlights it within the documents. Users can refine the model using simple mechanisms: click "x" next to words or documents to remove them, select and drag words to re-order them, type new words from the vocabulary into the input box and press "enter" to add them, select a word and click the trash can to add it to the stop words list, or click "split" and "merge" (to the right of the topic words) to enter into split and merge modes.

and codes clarified through discussion, and a second round of coding on transcripts for a different random participant achieved better agreement (researchers agreed on codes for 14 of 15 instances). One researcher then coded the remaining transcripts.

Findings: Simplicity and Improvement

We discuss findings related to refinement and save strategies, ability to improve the topic model, and challenges faced in using a fully functional HL-TM system.

Users prefer simple refinements

Like Lee et al. [27], simple refinements, such as *remove word*, *change word order*, and *add word to stop words* were the most commonly used. While perceived utility aligned with usage in Lee et al. [27], which is not surprising as refinements did not affect the model, there were two misaligned cases in our study: *change word order* and *add word* (Table 2). *Change word order* was the second most common refinement, yet only two of the 10 participants who used it in the task thought it was one of the most useful; alternatively, *add word* was only the fourth most common refinement, yet all six participants who used it thought it one of the most useful. These refinements provide varied control; we discuss this discrepancy in Discussion and Future Work.

Detailed refinements usage and strategies

We recorded which refinements participants used. The most common refinement, *remove word*, was used by 11 participants a total of 270 times, followed by *change word order* (10 participants, 136 times), *add to stop words* (seven participants, 90 times), and *add word* (six participants, 41 times). Other refinement operations were used by only three or fewer participants (Table 2).

When we asked participants the strategies they used, we got similar answers: *remove irrelevant words* (9 participants), *remove typos* (2), *skip bad topics* (2), *group common words* (2), *change word order to name* (2), *move irrelevant words to the end of the list* (1) , and *pinpoint refine* (1). To remove irrelevant words, participants were not consistent, instead employing both *remove word* and *add to stop words*. For example, P6 described that he would, *"first remove all similar words (e.g., make/makes) in each topic and then put all generic words in the stop words list."* Two participants described using *change word order* not only to fix the relative importance of words, but to name a topic, which they did by dragging three descriptive words to the front of the word list (each topic was represented by its top three words in the topic list on the left of the interface). A more expected usage of *change word order* came from P4, who said, *"I reordered the airline names to go to the end as I was not interested in what airlines attracted complaints."* For dealing with poor quality topics, two participants described their strategy to ignore bad topics, while one participant described a pinpoint refinement strategy in which she would choose a single topic word from a seemingly random topic and then use *add word* and *remove word* to make the topic more about that single word. Finally, we also noted cases of participants using refinements to explore the model. For example, P10 used the *add word* refinement to see if words showed up in the topic's documents, by first adding a word and then hovering over it to see it highlighted in the documents. P10 would then undo the added word if it did not appear in any of the top documents.

Table 1. Initial *test topic model* of 10 topics generated for the negative tweets from the airline Twitter corpus. Topics are represented by their top words. Observed topic coherence calculated by NPMI, which deems topics to be of higher quality if they contain words that appear more frequently together than apart in a reference corpus.

Topic ID	NPMI	Topic Words
T1	.031	hold, usairways, americanair, call, back, phone, hours, wait, change, minutes
T2	.014	southwestair, virginamerica, ticket, united, amp, fly, website, boarding, time, guys
T3	.024	flight, usairways, delayed, hrs, hours, late, miss, made, delay, connection
T4	.045	united, bag, bags, luggage, lost, baggage, check, find, airport, time
T5	.015	jetblue, http, time, united, email, long, jfk, give, amp, guys
T6	.029	americanair, usairways, people, weather, due, day, airport, hotel, issue, issues
T7	.022	united, plane, gate, waiting, hour, seat, sitting, crew, delay, min
T8	.009	usairways, americanair, make, problems, days, travel, refund, miles, told, booking
T9	.030	service, customer, united, usairways, worst, airline, experience, agents, staff, flying
T10	.025	flight, cancelled, southwestair, flightled, americanair, flights, today, flighted, late, tomorrow

Table 2. List of refinements ordered by in-task usage with count of participants that selected the specified refinement as one of the most useful or least useful refinements. Simple, word-level refinements are both the most commonly used and judged to be most useful (except for change word order: only two of the 10 participants who used it found it to be most useful).

Refinement	Most Useful	Least Useful	Used By	Total Usage
Remove word	5	1	11	270
Change word order	2	1	10	136
Add to stop words	3	0	7	90
Add word	6	1	6	41
Remove document	0	3	3	20
Merge topic	2	3	2	5
Split topic	1	5	1	1

Table 3. Save strategies described by participants and the number of times each participant saved during the task, ordered from most to least iterations. There was no dominant strategy: save usage and strategy varied across participants.

Participant	Iterations	Save Strategy
P8	42	After each refinement
P9	28	When sure
P12	19	After a batch of refinements
P2	19	After each topic modified
P7	18	After a batch of refinements
P10	16	After each refinement
P11	15	When sure
P4	9	After each topic modified
P5	8	After each refinement
P1	3	Forgot to save
P6	1	Forgot to save
P3	0	Afraid to save

When and why do users choose to save their changes?

Users refine the topic model by applying refinements and then separately clicking "save". Before saving, users can undo some or all of their changes. To understand when participants choose to save and because the HL-TM system does not support undo after saving, the system did not enforce a particular save strategy, such as after every refinement or a set number of refinements. Instead, participants could specify a series of local refinements, but these would only be applied to the model once they clicked "save", which they could do at any time. Save usage varied substantially ($min = 0$, $max = 42$, $avg = 14$, $SD = 12$); see Table 3.

Users were asked about strategies for when to click "save": *after each refinement* (4 participants), *after each topic modified* (2), *after a batch of refinements* (2), *when sure* (2). These varied strategies suggest that HL-TM should allow users to choose when to save their refinements. Additionally, two participants *forgot to save*, and another was *afraid to save*, which suggests that systems should remind users to save and support undo. "Save" counts and strategy feedback (Table 3).

P8 saved the most frequently (42 times) and described his strategy as saving after each refinement, saying, *"I always press the save button when I make any refinements."* P9 saved 28 times, saying, *"only when I am very sure about the result, I would press the save button."* In contrast, P6 and P1 reported that they forgot to save, and four other participants stated that remembering to save was one of the main challenges of using the tool. P12 wondered, *"if moving the [save] button over from the side would have helped me remember [to save]."* Finally, P3 was afraid to save, saying, *"I didn't want to start from scratch."* She suggested that having a history of refinements that could have been rolled back might mitigate timidity.

Did users improve the model?

To determine if participants improved the initial topic model using the HL-TM tool, we measured the quality of the initial topic model and the final topic models using qualitative and quantitative methods.

All participants started with the same model. We computed topic quality for the initial model and final models using a topic coherence metric based on NPMI [25]. The average topic coherence for the 10 topics of the initial model was .024 ($min = .01$, $max = .04$, $SD = .01$) (per-topic coherence shown in Table 1). The average topic coherence for the final model for each participant ranged from .021 to .037 ($M = .027$, $SD = .005$), which a paired t-test showed a significant improvement from the refinement process, $t(10) = 2.89$, $p = .037$.

Participants gave their satisfaction with the topic model before and after the task on a scale from one to seven, with one being not at all satisfied and seven being very satisfied. The average subjective model satisfaction increased from 4.7 ($SD = 1.29$) before the task to 5.2 ($SD = 0.83$) after the task. While this increase was not statistically significant by a Wilcoxen signed rank test ($Z = -1.04$, $p = .15$), six of the 12 participants commented unprompted after the task that their final model provided a good organization of the complaints. For example, P5 said, *"I'm overall happy with the [final] model and I like that I can use the tool to make the changes that I want."*

Participants gave the best and worst topics in the initial model (Table 1). Most participants agreed the best topics were T4 (4 participants), T3 (3), T1 (3), and T9 (3) and the worst topics were T5 (8), T6 (3), and T8 (2), which correlates with the observed topic coherence. The three best topics by NPMI are T4 (NPMI=.045), T1 (.031), and T9 (.030), while the three worst topics are T8 (.009), T2 (.014), and T5 (.015).

What challenges do users face?

To understand how general interactive machine learning challenges affect users of HL-TM, we coded four common challenges—tracking complex changes, instability, lack of control, latency—and identified challenges with our system. Participants also stated which challenges were most and least frustrating during the task. Of the four common challenges, tracking complex changes was the most frustrating, followed by instability, lack of control, and latency was the least frustrating challenge.

Tracking complex changes

When users click "save", the algorithm updates the model, and the resulting model may have substantial changes. To explore whether participants could track these changes, they rated their agreement with the statement, *"I was able to remember what the model looked like before my updates"* on a scale from one, meaning no agreement, to seven, meaning complete agreement (Figure 2, D) and discussed how this affected them. The average response was 3.7 out of 7 ($SD = 1.7$), and four of the 12 participants said this was the most frustrating challenge while one said it was the least.

Five participants said not being able to remember what the model looked like hurt their performance. For example, P9 said, *"a moment ago, I was satisfied with this topic, but now it's gone, and I don't think I am, but I can't remember,"* and P3 and P8 felt the lack of "undo" intensified this challenge. P3 said, *"I think this is a big issue–I'd like to know if I'm capturing the true data–and be able to step back to early versions of the model before saving,"* and P8 said, *"I don't know what I have done sometimes, and there are no ways to go back . . . "*. Four participants mentioned a similar challenge, that it was hard to tell what changed in the model after an update, such as P10, who *"had to brush through all the words to confirm if [his specified] change occurred,"* and P5, who *"did not understand it at first, that the model actually changes, as there was no feedback or indication."* Finally, three participants requested a long-term history view of the model, such as P3, who suggested *"having a history of refinements."*

Stability

A user interface should be predictable to support user confidence and understanding [18]; however, interactive machine learning often violates this principle [2] as these systems combine hidden knowledge—such as previously learned models or data—with users' instructions. We describe system predictability as a combination of stability (no changes other than those specified occur) and control (the change occurs as specified).

We asked if participants agreed with the statement, *"no changes other than the refinements I made occurred when I clicked update"* on a scale from one, or no agreement, to seven, or complete agreement (Figure 2, C). The average response was 4.1 out of seven ($SD = 2.0$), and three of 12 participants said instability was the most frustrating challenge while no participants said it was the least.

There was a large variance for not only whether users perceived instability but also their reactions to it. After the task, eight of 12 participants mentioned they had perceived instability. Of those, two participants found this to be positive. For example, P6 observed an unspecified change when *"new words were added on to the list to replace the ones I removed. It made the model better."* P2 noted that after removing some words from a topic there was *"some slight surprise at seeing words that I had not chosen show up, but I was pretty satisfied on looking at the results."* Three participants felt neutral about the instability. For example, P7 said, *"[instability] did not impact"* his ability to perform the task, and P4 said, *"when I removed some keywords, other keywords came up. I wasn't paying enough attention to this to determine if it helped or harmed."* Finally, three participants had negative reactions, such as P9, who was unsure of what had changed in the model after an update, but stated, *". . . but I remember being happy with the topic and when that changed it made me unhappy."* This participant also requested the ability to *freeze a topic*, meaning it would not be changed as other refinements were made.

Control

Similar to the challenge of instability, users should always be in control of user interfaces [34]; however, interactive machine learning is by definition a *collaboration* between algorithm and user [19]. In this collaboration, we define "control" as whether the user's refinements are incorporated into the model as expected. For example, if a user removes a word from a topic, the word should not be in the topic after the update.

To explore whether participants felt in control of the system, they stated on a seven-point scale whether they agreed with the statement, *"the refinements I made were applied as expected when I clicked update"* from not agreeing at all to completely agreeing (Figure 2, B) and discussed how this affected their task. The average response was 5.6 out of seven ($SD = 1.0$), meaning overall users found the system to be fairly controllable. One of the 12 participants said lack of control was the most frustrating challenge and one said it was the least.

However, during the task seven participants noted frustration with the lack of control with the interface, and five participants

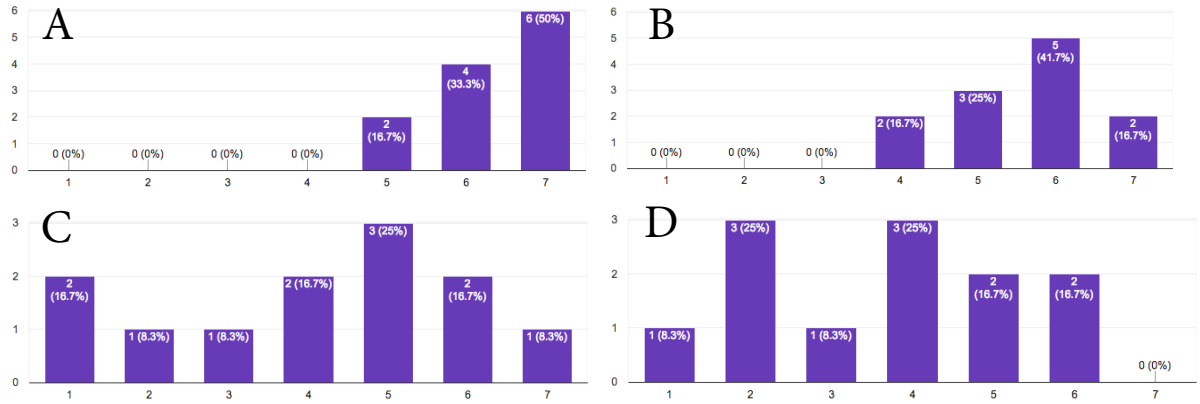

Figure 2. Counts for responses on a scale from one to seven for participants' agreement with statements related to latency (A), lack of control (B), instability (C), and tracking complex changes (D), with seven meaning they did not experience it and one that they did. Most participants found that the system updated quickly and refinements were applied as expected, while there was substantial variance for if participants could remember what the model looked like before updating or if they felt the updated model included other changes than those specified.

specifically observed that *change word order* was uncontrollable. P4 tried to drag important words to the front of the topic list and stated that, *"the reordering didn't always get accepted,"* and P8 tried to drag unimportant words to the end of the list and said, *"I tried to move this word and it just goes back up."*

Latency

Prior work in interactive machine learning calls for rapid interaction cycles [2] to minimize attention loss [22] and reduce short-term memory load [34]. However, many interactive systems do not provide real-time updates: this latency—the user has to wait while the algorithm is performing an update—is typically related to the size of the data and complexity of the computation. For example, an early HL-TM implementation [23] took 5 to 50 seconds to update the model based on refinement operations.

Our refinement implementation is efficient by design, and the data set used in this user study was relatively small (both in document size and length), therefore the algorithm updated almost instantaneously during the task (.09 – .63 seconds). No participant said that latency was the most frustrating challenge while two participants said it was the least frustrating, and the average response was 6.3 out of seven ($SD = 0.8$) for participants agreement with the statement, *"after clicking the update button, the model updated quickly"* (Figure 2, A).

However, for a more realistic corpus size or alternative refinement implementation, latency becomes a challenge. We asked participants to describe how their ability to perform the task would be affected had the wait time been 10 seconds, 30 seconds, two minutes, or 10 minutes. Most participants felt 10 seconds would be an acceptable time to wait: five participants felt that waiting 10 seconds would have no effect on the task and two participants felt that this longer wait time would have a positive effect, for example, P5 stated that waiting longer *"would be better for me to realize that the tweets have changed."* For a 30 second wait time, two participants felt this would be an acceptable wait time without any changes to the interface, whereas four participants said that changes to the interface would be required for this longer wait time. P7 worried this wait would further hinder the ability to remember what the model looked like before updating, and P3 thought this would further affect save strategy, suggesting that it would instead be *"better to not 'save' changes, but to have highlights to show what it 'might' look like once saved."* Most participants felt that both two minutes and 10 minutes would be unacceptable wait times.

Trust and confidence

Trust is a primary design challenge for intelligent systems [32, 20]. An entire sub-field of machine learning focuses on interpretability to build trust and promote adoption of these systems. Surprisingly, we did not see participants mistrusting the HL-TM system. While users are quick to distrust a classification system that produces easily identifiable, incorrect classifications, topic models have less obvious *incorrect* answers.

However, participants sometimes put too much trust in the system or lacked self-confidence. For example, P10 was confused about a topic word, saying, *"if the system coughed it up, there must be a reason for it, right?"* Some participants lacked confidence in their refinements: P7 said that *remove document* is the least useful refinement, because, *"I don't feel comfortable removing a document."* And when P5 added words to a topic, she said, *"it's putting my words on top . . . I've added too many words, which have gone to the top of the list, so either the algorithm thinks it's important or it's because I've added them,"* followed by, *"I don't think that it should always give more importance [to my added words], because I could be wrong!"* This challenge has a direct connection to the issues of instability and lack of control, which we discuss in more detail in Discussion and Future Work.

What other requests do users have for the system?

Many participants wanted a better understanding of the model and the data. For example, two participants requested a better model overview, such as P7, who wanted to *"see the entire list of the top 20 words for each topic on one screen to allow for*

making bulk, faster changes." Additionally, three participants wanted to view words or documents across topics, such as P12 who suggested, *"a note or color to indicate that a certain term appears only in this topic and not in the others."* Two participants requested enhancing the word in context feature, such as by scrolling to the selected word or filtering to only documents containing the word. Three participants wanted to view more documents than the 40 shown, and two participants wanted to view the total number of documents for a topic.

Similar to the *merge word* operation identified by Lee et al. [27], six participants requested a refinement to add phrases (instead of just single words), and four participants requested a refinement to group synonyms and plurals. As anticipated, participants used the *add to stop words* refinement, and two participants requested an enhancement to the stop words functionality, such as being able to view the stop words list and remove words that have been added to it. However, seven participants noted confusion between the *add to stop words* and *remove word* refinements, which should be clarified in future interface design. For example, P5 said, *"removing a word feature is similar to the delete feature, which got me a bit confused,"* and P9 said, *"I got confused between removing keywords from a particular [topic] and the overall [topics], so I made mistakes in the beginning."* To help better organize the view, three participants wanted to name topics, noting that it would be a useful way to remember what the topics are about, and two participants wanted to reorder topics in the list. Finally, two participants wanted to delete a topic if it was particularly bad.

Summary

Participants were frustrated by their inability to track how the model changed throughout the refinement process. While participants perceived system instability, they had varied reactions (positive and negative). On the other hand, users did not experience substantial latency or lack of control. We did not find any cases where users distrusted the system, but users perhaps trusted the system too much or had too little confidence in themselves. Participants specifically requested the ability to undo changes after saving and to curate the topic model view, such as by re-ordering the topic list, removing poor quality topics, and naming topics. Participants also requested multi-word refinements, such as adding phrases and grouping synonyms.

DISCUSSION AND FUTURE WORK

We outline implications for future HL-TM system design, discuss open questions related to interactive machine learning, and provide a reflection on our HL-TM implementation.

Design Recommendations

Provide richer history: Participants voiced concerns with their inability to remember the history of the model, and four of 12 participants said they were unable to tell how the model has changed after an update. HL-TM interfaces should strive to support visualization of short term and long term model changes; users want to track how the model changed throughout the refinement process. This was the most consistent and most frustrating issue in the study.

Support undo: When possible, HL-TM should support reverting to prior states of the model: some noted that this made them afraid to save during the task, while others specifically requested an undo functionality.

Allow users to choose when to save, but remind them to do so: We had anticipated needing a separate save action to allow users to confirm refinements (lacking undo) and to counteract latency, but we also noted users who created refinements as a data exploration tool without the intent of having them update the model. Thus, HL-TM systems should allow users to choose when to save their refinements to the model instead of forcing a save. However, because users forget to save, additional information should be provided in the interface to remind users, such as a more prominent count of outstanding refinement operations or a visual cue that displays if they have not saved recently.

Freeze topics to protect from instability: Users complained of instability when topics that were once high quality or about a particular thing had changed. A process, such as freezing a topic, suggested by one participant as a mechanism to hold a particular topic constant during subsequent updates, is a promising solution to this problem and should be incorporated in future design.

Support multi-word refinements: Participants requested the ability to add phrases and group synonyms. Group synonyms could be implemented as the merge word refinement discussed in Lee et al. [27], not as an update to the underlying model, but as a way of organizing words in the interface. On the other hand, add phrases should be implemented in the interface as an extension to *add word* (as requested by participants), but would likely require a more complex modeling approach that supports n-grams as opposed to single tokens.

Clarify difference between adding a word to stop words and removing it from a single topic: Future design should explicitly delineate between removing a word from all topics (and the modeling process entirely), *add to stop words*, and removing a word from a single topic, *remove word*, as many participants confused the two operations during the task.

Support user-curated model view: Three participants requested named topics. Two other participants used *change word order* for *ad hoc* topic naming. As this operation is not always applied as expected, providing a controllable topic naming functionality will improve user experience. Participants also requested other techniques for curating their model view, which should be incorporated in the design of future systems, such as the ability to re-order the topic list and to remove poor quality topics entirely.

Open Questions

This is the first system to efficiently implement a full suite of refinements desired by users in prior work [27, 31], enabling the study of true human-in-the-loop interactions of a comprehensive HL-TM system. We enumerate open questions about HL-TM design that follow from our findings.

Trust vs. instability and control: Users were not bothered by instability or lack of control either because they trusted

the system or had little confidence in themselves. Specifically, users with limited confidence blamed *themselves* for creating poor refinements (i.e., when the change did not happen as anticipated). If system builders do not want novice users to feel like the "junior partner" in the human-machine collaboration, future work should explore whether ensuring users understand the teaming aspect of these systems can improve their experience and make unpredictability more acceptable (and sometimes welcome, as it can drive discovery).

Trust, control, and refinement: Lee et al. [27] studied refinement usage without a refinement implementation, meaning users did not see the full effect of their refinements on the model. In that study, *remove document* was a commonly used refinement, however, that is not the case in our study. Before the study participants worried that it may take too long to determine which documents to remove, while afterwards noted they lacked confidence to remove a document. Although Lee et al. [27] considered that refinements that take too long would hurt usage, lack of trust or confidence in HL-TM is a new challenge to consider.

Change word order was commonly used, but frustrating to users, while *add word* was used less, yet all participants who used it thought it was useful. This discrepancy highlights the difference in control of the two refinements: *change word order* was unpredictable and thus frustrating, but *add word* always worked on the first try.

Save strategy and instability: When users save after a batch of refinements (as opposed to a single requirement) their intentions are clearer. This in turn minimizes instability as the system has more information to incorporate into the model. On the other hand, each refinement may have cascading effects, and a batch of refinements could therefore appear to be more unstable than a single refinement. We did not find a relationship between users' described save strategies and their perceived system instability. Future work should explore the relationship with a specific focus on how much information users provide to the system and whether this information affects the system's stability and how users react.

Algorithm Reflection

This work proposes an asymmetric prior-based HL-TM implementation. We implemented seven refinement operations using the proposed algorithm, which can be easily extended to other refinements, such as creating a new topic using seed words or deleting a topic. One limitation of this algorithm is the difficulty to specify word order constraints. For example, if a user wants to change a word's position from rank eight to two in the word list, the algorithm cannot reliably maintain the exact user provided word order. We argue that topic models are probabilistic models and during parameter estimation they can ignore user provided feedback if the underlying data does not support the user's hypothesis. For example, if a user wants to add a word to a topic that only shows up a few times in the corpus, the model might not put that word in the list of top ranked words for that topic. Another limitation of our algorithm is with the *split topic* refinement; our proposed implementation cannot reliably generate a good quality topic if the user provides only very few or unrelated seed words.

CONCLUSION

Prior work in HL-TM either implemented refinement operations without first understanding the needs of end users [23, 21, 10] or identified the refinement operations that users wish to do [27], but did not implement them. This work is the first to examine user experience with a fully-functional HL-TM system that contains the refinements users want. Specifically, we validate prior results, such as refinement usage and effectiveness, and explore how these and user experience are affected by previously hidden issues, such as unpredictability, trust, and lack of control. We also present suggestions, such as the need to visualize complex model changes and support undo. Non-expert end users used the system to refine a topic model and we explored how these users perceived and were affected by common challenges in interactive machine learning, such as latency, unpredictability, and trust. Participants improved a topic model using the tool and identified additional refinement and tool suggestions that should guide HL-TM development.

Acknowledgments

This work was supported by the collaborative NSF Grant IIS-1409287 (UMD) and IIS-1409739 (BYU). Boyd-Graber is also supported by NSF grant IIS-1652666 and IIS-1748663. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

REFERENCES

1. Eric Alexander and Michael Gleicher. 2016. Assessing Topic Representations for Gist-Forming. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. 100–107. DOI: http://dx.doi.org/10.1145/2909132.2909252
2. Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the People: The Role of Humans in Interactive Machine Learning. (2014). DOI: http://dx.doi.org/10.1609/aimag.v35i4.2513
3. David Andrzejewski, Xiaojin Zhu, and Mark Craven. 2009. Incorporating Domain Knowledge into Topic Modeling via Dirichlet Forest Priors.. In *Proceedings of the International Conference on Machine Learning*, Vol. 382. 25–32. DOI: http://dx.doi.org/10.1145/1553374.1553378
4. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research* 3, 1 (2003), 993–1022. DOI: http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993
5. G. Bouma. 2009. Normalized (Pointwise) Mutual Information in Collocation Extraction. In *Proceedings of the Biennial GSCL Conference*. Tübingen, 31–40.
6. Jordan Boyd-Graber, David M. Blei, and Xiaojin Zhu. 2007. A Topic Model for Word Sense Disambiguation.
7. Jordan Boyd-Graber, Yuening Hu, and David Mimno. 2017. *Applications of Topic Models*. Foundations and Trends in Information Retrieval, Vol. 11. NOW Publishers. http://www.nowpublishers.com/article/Details/INR-030

8. Virginia Braun and Victoria Clarke. 2006. Using Thematic Analysis in Psychology. *Qualitative Research in Psychology* 3, 2 (1 2006), 77–101. DOI: http://dx.doi.org/10.1191/1478088706qp063oa
9. Allison June-Barlow Chaney and David M. Blei. 2012. Visualizing Topic Models. In *Proceedings of the International Conference on Weblogs and Social Media*. 419–422.
10. Jaegul Choo, Changhyun Lee, Chandan K. Reddy, and Haesun Park. 2013. UTOPIAN: User-Driven Topic Modeling Based on Interactive Nonnegative Matrix Factorization. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 1992–2001. DOI: http://dx.doi.org/10.1109/TVCG.2013.212
11. Jason Chuang, Yuening Hu, Ashley Jin, John D. Wilkerson, Daniel A. McFarland, Christopher D. Manning, and Jeffrey Heer. 2013. Document Exploration with Topic Modeling: Designing Interactive Visualizations to Support Effective Analysis Workflows. In *Conference on Neural Information Processing Systems (NIPS) Workshop on Topic Models: Computation, Application, and Evaluation*. 1–4.
12. Fan Du, Catherine Plaisant, Neil Spring, and Ben Shneiderman. 2017. Finding Similar People to Guide Life Choices: Challenge, Design, and Evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. DOI: http://dx.doi.org/10.1145/3025453.3025777
13. Jacob Eisenstein, Duen Horng Chau, Aniket Kittur, and Eric Xing. 2012. TopicViz: Interactive Topic Exploration in Document Collections. In *Proceedings of the ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts*. 2177–2182. DOI:http://dx.doi.org/10.1145/2212776.2223772
14. Krzysztof Z. Gajos, Katherine Everitt, Desney S. Tan, Mary Czerwinski, and Daniel S. Weld. 2008. Predictability and Accuracy in Adaptive User Interfaces. In *Proceeding of the SIGCHI conference on Human Factors in Computing systems*. 1271–1274. DOI: http://dx.doi.org/10.1145/1357054.1357252
15. Matthew J. Gardner, Joshua Lutes, Jeff Lund, Josh Hansen, Dan Walker, Eric Ringger, and Kevin Seppi. 2010. The Topic Browser: An Interactive Tool for Browsing Topic Models. In *Proceedings of the Workshop on Challenges of Data Visualization, held in conjunction with the 24th Annual Conference on Neural Information Processing Systems*. 1–9. http://cseweb.ucsd.edu/~lvdmaaten/workshops/nips2010/papers/gardner.pdf
16. Derek Greene and Padraig Cunningham. 2006. Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering. In *Proceedings of the International Conference on Machine Learning*. 377–384. DOI:http://dx.doi.org/10.1145/1143844.1143892
17. Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 101. 5228–35. DOI: http://dx.doi.org/10.1073/pnas.0307752101
18. Robert Hoekman. 2007. *Designing the Obvious: A Common Sense Approach to Web Application Design*. Vol. 48. 255 pages. http://www.worldcat.org/isbn/032145345X
19. Lars Erik Holmqvist. 2017. Intelligence on tap: artificial intelligence as a new design material. *Interactions* (2017), 28–33. DOI: http://dx.doi.org/https://doi.org/10.1145/3085571
20. K. Höök. 2000. Steps to take before intelligent user interfaces become real. *Interacting with Computers* 12, 4 (2000), 409–426. DOI: http://dx.doi.org/10.1016/S0953-5438(99)00006-5
21. Enamul Hoque and Giuseppe Carenini. 2015. ConVisIT: Interactive Topic Modeling for Exploring Asynchronous Online Conversations. In *Proceedings of the International Conference on Intelligent User Interfaces*. ACM, New York, 169–180. DOI: http://dx.doi.org/10.1145/2678025.2701370
22. Eric Horvitz. 1999. Principles of Mixed-Initiative User Interfaces. In *Proceeding of the SIGCHI conference on Human Factors in Computing Systems*. ACM, New York, 159–166. DOI:http://dx.doi.org/10.1145/302979.303030
23. Yuening Hu, Jordan Boyd-Graber, Brianna Satinoff, and Alison Smith. 2014. Interactive Topic Modeling. *Machine Learning* 95, 3 (6 2014), 423–469. DOI: http://dx.doi.org/10.1007/s10994-013-5413-0
24. Antti Kangasrääsiö, Dorota Głowacka, and Samuel Kaski. 2015. Improving Controllability and Predictability of Interactive Recommendation Interfaces for Exploratory Search. In *Proceedings of the International Conference on Intelligent User Interfaces*. 247–251. DOI: http://dx.doi.org/10.1145/2678025.2701371
25. Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine Reading Tea Leaves : Automatically Evaluating Topic Coherence and Topic Model Quality. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*. 530–539. http://www.aclweb.org/anthology/E14-1056
26. Hanseung Lee, Jaeyeon Kihm, Jaegul Choo, John Stasko, and Haesun Park. 2012. iVisClustering: An Interactive Visual Document Clustering via Topic Modeling. *Computer Graphics Forum* 31, 3pt3 (2012), 1155–1164. DOI: http://dx.doi.org/10.1111/j.1467-8659.2012.03108.x
27. Tak Yeon Lee, Alison Smith, Kevin Seppi, Niklas Elmqvist, Jordan Boyd-Graber, and Leah Findlater. 2017. The Human Touch: How Non-ExpertUusers Perceive, Interpret, and Fix Topic Models. *International Journal of Human Computer Studies* 105 (2017), 28–42. DOI: http://dx.doi.org/10.1016/j.ijhcs.2017.03.007

28. Jeff Lund, Connor Cook, Kevin Seppi, and Jordan Boyd-Graber. 2017. Tandem Anchoring: A Multiword Anchor Approach for Interactive Topic Modeling. In *Association for Computational Linguistics*.

29. Theo Mandel. 1997. *The Elements of User Interface Design*. 440 pages. http://books.google.com/books?id=H-ZQAAAAMAAJ&pgis=1

30. David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing Semantic Coherence in Topic Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 262–272.

31. Chris Musialek, Philip Resnik, and Andrew S. Stavisky. 2016. Using Text Analytic Techniques to Create Efficiencies in Analyzing Qualitative Data : A Comparison between Traditional Content Analysis and a Topic Modeling Approach. In *AAPOR Conference*.

32. Donald A Norman. 1994. How might people interact with agents. *Commun. ACM* 37, 7 (1994), 68–71. DOI: http://dx.doi.org/10.1145/176789.176796

33. Amir M. Saeidi, Jurriaan Hage, Ravi Khadka, and Slinger Jansen. 2015. ITMViz: Interactive Topic Modeling for Source Code Analysis. In *Proceedings of the IEEE International Conference on Program Comprehension*. 295–298. DOI:http://dx.doi.org/10.1109/ICPC.2015.44

34. Ben Shneiderman, Catherine Plaisant, Maxine Cohen, and Steven Jacobs. 2009. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Vol. 25. 624 pages.

35. Alison Smith, Tak Yeon Lee, Forough Poursabzi-Sangdeh, Jordan Boyd-Graber, Niklas Elmqvist, and Leah Findlater. 2017. Evaluating Visual Representations for Topic Understanding and Their Effects on Manually Generated Topic Labels. *Transactions of the Association for Computational Linguistics* 5 (2017), 1–15.

36. Hanna M Wallach, David Mimno, and Andrew Mccallum. 2009. Rethinking LDA : Why Priors Matter. *Advances in Neural Information Processing Systems 22* 22, 2 (2009), 1973–1981. DOI: http://dx.doi.org/10.1007/s10708-008-9161-9