Why Didn't You Listen to Me? Comparing User Control of Human-in-the-Loop Topic Models

Varun Kumar [*]
Amazon Alexa
Cambridge, MA
varunk@cs.umd.edu

Alison Smith-Renner
University of Maryland
College Park, MD
amsmit@cs.umd.edu

Leah Findlater
University of Washington
Seattle, Washington
leahkf@uw.edu

Kevin Seppi
Brigham Young University
Provo, UT
kseppi@gmail.com

Jordan Boyd-Graber
University of Maryland
College Park, MD
jordanbg@umiacs.edu

Abstract

To address the lack of comparative evaluation of Human-in-the-Loop Topic Modeling (HLTM) systems, we implement and evaluate three contrasting HLTM modeling approaches using simulation experiments. These approaches extend previously proposed frameworks, including constraints and informed prior-based methods. Users should have a sense of control in HLTM systems, so we propose a *control* metric to measure whether refinement operations' results match users' expectations. Informed prior-based methods provide better control than constraints, but constraints yield higher quality topics.

1 Human-in-the-Loop Topic Modeling

Topic models help explore large, unstructured text corpora by automatically discovering the topics discussed in the documents (Blei et al., 2003). However, generated topic models are not perfect; they may contain incoherent or loosely connected topics (Chang et al., 2009; Mimno et al., 2011; Boyd-Graber et al., 2014).

Human-in-the-Loop Topic Modeling (HLTM) addresses these issues by incorporating human knowledge into the modeling process. Existing HLTM systems expose topic models as their topic words and documents, and users provide feedback to improve the models using varied refinement operations, such as adding words to topics, merging topics, or removing documents (Smith et al., 2018; Wang et al., 2019). Systems also vary in how they incorporate feedback, such as "must-link" and "cannot-link" constraints (Andrzejewski et al., 2009; Hu et al., 2014), informed priors (Smith et al., 2018), or document labels (Yang et al., 2015). However, evaluations of these systems are either not comparative (Choo et al., 2013; Lee et al., 2017) or compare against non-interactive models (Hoque and Carenini, 2015; Hu et al., 2014) or for only a limited set of refinements (Yang et al., 2015; Xie et al., 2015). Evaluations are thus silent on which HLTM system best supports users in improving topic models: they ignore whether refinements are applied correctly or how they compare with other approaches. Moreover, comparative evaluations can be difficult because existing HLTM systems support diverse refinement operations with little overlap.

To address these issues, we implement three HLTM systems that differ in the techniques for incorporating prior knowledge (informed priors vs. constraints) and for inference (Gibbs sampling vs. variational EM), but that all support seven refinement operations preferred by end users (Lee et al., 2017; Musialek et al., 2016). We compare these systems through experiments simulating random and "good" user behavior. The two Gibbs sampling-based systems extend prior work (Yang et al., 2015; Smith et al., 2018), but to our knowledge, the combination of informed priors and variational inference in an HLTM system is new. Additionally, while Yang et al. incorporate word correlation knowledge and document label knowledge into topic models, this paper extends their modeling approach with the implementation of seven new user refinements.

[*]Work performed at University of Maryland, College Park

We also introduce metrics to assess the degree to which HLTM systems listen to users—*user control*—a key user interface design principle for human-in-the-loop systems (Amershi et al., 2014; Du et al., 2017). In general, informed priors provide more control while constraints produce higher quality topics.

This paper provides three contributions: (1) implementation of an HLTM system using informed priors and variational inference, (2) experimental comparison of three HLTM systems, and (3) metrics to evaluate user control in HLTM systems.

2 Human Feedback and LDA

We briefly describe Latent Dirichlet Allocation (Blei et al., 2003, LDA) and outline the experimental conditions and our implementation.

2.1 LDA Inference

LDA is generative, modeling documents as mixtures of k topics where each topic is a multinomial distribution, ϕ_z, over the vocabulary, V. Each document d is an admixture of topics θ_d. Each word indexed by i in document d is generated by first sampling a topic assignment $z_{d,i}$ from θ_d and then sampling a word from the corresponding topic ϕ_{z_i}.

Collapsed Gibbs sampling (Griffiths and Steyvers, 2004) and variational Expectation-Maximization (Blei et al., 2003, EM) are two popular inference methods to compute the posterior, $p(z, \phi, \theta \,|\, w, \alpha, \beta)$. Gibbs sampling iteratively samples a topic assignment, $z_{d,i} = t$ given an observed token $w_{d,i}$ in document d and other topic assignments, $z_{-d,n}$, with probability

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

Here, $n_{d,t}$ is the count topic t is in document d, $n_{w,t}$ is the count of token w in topic t, and n_t is the marginal count of tokens assigned to topic t. Alternatively, variational EM approximates the posterior using a tractable family of distributions by first defining a mean field variational distribution

$$q(z, \phi, \theta \,|\, \lambda, \gamma, \pi) = \prod_{k=1}^{K} q(\phi_k \,|\, \lambda_k) \prod_{d=1}^{D} q(\theta_d \,|\, \gamma_d) \prod_{n=1}^{N_d} q(z_{dn} \,|\, \pi_{dn}) \quad (2)$$

where γ_d, π_d are local parameters of the distribution q for document d, and λ is a global parameter. Inference minimizes the KL divergence between the variational distribution and true posterior. While there are many LDA variants for specific applications (Boyd-Graber et al., 2017), we focus on models that interactively refine initial topic clustering.

2.2 HLTM Modeling Approaches

To investigate adherence to user feedback and topic quality improvements, we compare HLTM systems, based on three modeling approaches. Each of these approaches incorporate user feedback by first *forgetting* what the model learned before, by unassigning words from topics (Hu et al., 2014), and then *injecting* new information based on user feedback into the model.

We compare two existing techniques for *injecting* new information: (1) asymmetric priors (or informed priors), which are used extensively for injecting knowledge into topic models (Fan et al., 2017; Zhai et al., 2012; Pleplé, 2013; Smith et al., 2018; Wang et al., 2019) by modifying Dirichlet parameters, α and β, and (2) constraints (Yang et al., 2015), in which knowledge source m is incorporated as a potential function $f_m(z, m, d)$ of the hidden topic z of word type w in document d. While other frameworks exist (Foulds et al., 2015; Andrzejewski et al., 2009; Hu et al., 2014; Xie et al., 2015; Roberts et al., 2014), we focus on informed priors and constraints, as these are flexible to support the refinement operations preferred by users and reasonably fast enough to support "rapid interaction cycles" required for effective interactive systems (Amershi et al., 2014).

We also compare two inference techniques for topic models (1) Gibbs sampling and (2) variational EM inference. Because HLTM requires *forgetting* existing topic assignments (Hu et al., 2014), we use two different methods to forget existing topic assignments. In Gibbs sampling, information is forgotten by adjusting topic-word assignments, z_i. In variational EM, $\lambda_{t,w}$ encodes how closely the word w is related to topic t. In the *E-step*, the model assigns latent topics based on the current value of λ, and in the *M-step*, the model updates λ using the current topic assignments. Because the model relies on a fixed λ for topic assignment, information for a word w in a topic t can be forgotten by resetting $\lambda_{t,w}$ to the prior $\beta_{t,w}$. Together, these injection and inference techniques result in three HLTM modeling approaches:

Informed priors using Gibbs sampling (*info-gibbs*) forgets topic-word assignments z_i and injects new information by modifying Dirichlet parameters, α and β. Smith et al. (2018) implement seven refinements for this approach. We extend their work with a create topic refinement.

Informed priors using variational inference (*info-vb*) forgets topic-word assignments for a word w in topic t by resetting the value of $\lambda_{t,w}$. This approach manipulates priors, α and β, to incorporate new knowledge like *info-gibbs*. We define and implement seven *user-preferred* refinement operations for this approach.

Constraints using Gibbs sampling (*const-gibbs*) forgets topic assignments like in *info-gibbs*, but instead of prior manipulation, injects new information into the model using potential functions, $f_m(z, m, d)$ (Yang et al., 2015). We define and implement seven *user-preferred* refinement operations for this approach.

2.3 Refinement Implementations

Our three systems support the following seven refinements that users request in HLTM systems (Musialek et al., 2016; Lee et al., 2017):

Remove word w from topic t. For all three systems, first forget all w's tokens w_i from t. Then, for *info-gibbs* and *info-vb*, assign a very small prior[1] ϵ to w in t. For *const-gibbs*, add a constraint[2] $f_m(z, w, d)$, such that $f_m(z, w, d) = log(\epsilon)$ if $z = t$ and $w = x$, else assign 0.

Add word w to topic t. For all three systems, first forget w from all other topics. Then, for *info-gibbs* and *info-vb*, increase the prior of w in t by the difference between the topic-word counts of w and topic's top word $\hat{w}$ in t. For *const-gibbs*, add a constraint $f_m(z, w, d)$, such that $f_m(z, w, d) = 0$ if $z = t$ and $w = x$, else assign $log(\epsilon)$.

Remove document d from topic t. For all models, first forget the topic assignment for all words in the document d. Then, for *info-gibbs* and *info-vb*, overwrite the previous prior value with a very small prior ϵ, to t in α_d. For *const-gibbs*, add a constraint $f_m(z, w, d)$, such that $f_m(z, w, d) = log(\epsilon)$ if $z = t$ and $d = x$, else assign 0.

[1]We use $\epsilon = 10^{-8}$

[2]We use $log(\epsilon)$ to make it a soft constraint. Replacing it with -∞ will make it a hard constraint.

Merge topics t_1 and t_2 into a single topic, t_1. For *info-gibbs* and *const-gibbs*, assign t_1 to all tokens previously assigned to t_2. This effectively removes t_2 and updates t_1, which should represent both t_1 and t_2. For *info-vb*, add counts from λ_{t_2} to λ_{t_1} and remove row from λ corresponding to t_2.

Split topic t given seed words s into two topics, t_n, containing s, and t, without s. For each vocabulary word, move a fraction of probability mass from t to t_n as proposed by (Pleplé, 2013). Then, for *info-gibbs* and *info-vb*, assign a high prior for all s in t_n. Following Fan et al., we use 100 as the high prior. For *const-gibbs*, to s to t_n, add a constraint $f_m(z, w, d)$, such that $f_m(z, w, d) = 0$ if $z = t_n$ and $w = w_i \in s$, else assign $log(\epsilon)$.

Change word order , such that w_2 is higher than w_1 in topic t. In *info-gibbs*, increase the prior of w_2 in t by the topic word counts' difference $n_{w_{1,t}}$ -$n_{w_{2,t}}$. In *info-vb*, increase the prior by $\lambda_{t,w_1} - \lambda_{t,w_2}$. For *const-gibbs*, compute the ratio r between the topic word counts' difference $n_{w_{1,t}} - n_{w_{2,t}}$ and the counts of word w_2, which have any topic except t, $n_{w_{2,x,x\neq t}}$. Then, add a constraint $f_m(z, w, d)$, such that $f_m(z, w, d) = 0$ if $z = t$ and $w = w_2$, else assign δ where $\delta = \log(\epsilon)$ if $r > 1$ else $\delta = 1.0 - r$.

Create topic t_n, given seed words, s. First forget the topic assignment for all s. Then, for *info-gibbs* and *info-vb*, assign a high prior to s. For *const-gibbs*, to assign s to t_n, add a constraint $f_m(z, w, d)$, such that $f_m(z, w, d) = 0$ if $z = t_n$ and $w = w_i \in s$, else assign $log(\epsilon)$.

3 Measuring Control

Prior work in interactive systems emphasizes the importance of doing what users ask, that is, *end user control* (Shneiderman, 2010; Amershi et al., 2014). However, HLTM, which must balance modeling the data well and fulfilling users' desires, can frustrate users when refinements are not applied as expected (Smith et al., 2018). Evaluation metrics such as topic coherence, perplexity, and log-likelihood measure how well topics model data, but are not sufficient to measure whether user feedback is incorporated as expected. Therefore, we propose new *control* metrics to measure how well models reflect users' refinement intentions.

Consider a topic, t, as a ranked word list sorted in descending order of their probabilities in t. Let

$r_{w_t}^{M_1}$ denote the rank of a word w in topic t in model M_1. After applying a word-level refinement, the rank of w in the updated model M_2, is $r_{w_t}^{M_2}$. For word-level refinements, such as **add word**, **remove word**, and **change word order**, compute *control* as the ratio of the actual rank change, the absolute difference ($r_{w_t}^{M_1} - r_{w_t}^{M_2}$), and the expected rank change. A score of 1.0 indicates that the model perfectly applied the refinement, while a negative score indicates the model did the opposite of what was desired. For **remove document**, use the same definition as **remove word** except consider a topic as a ranked document list.

For **create topic**, compute *control* as the ratio of the number of seed words in the created topic out of the total number of provided seed words. For **merge topics**, *control* is defined as the ratio of the number of words in the merged topic which came from either of the parent topics, and the total number of words shown to a user. For **split topic**, *control* is the average of the *control* scores of parent topic and child topic, computed using the *control* definition for **create topic**.

4 HLTM System Comparison

To compare how the three HLTM systems model data and adhere to user feedback (i.e., provide control), we need user data; however, real user interaction is expensive to obtain. So, we simulate a range of user behavior with these systems: users that aim to improve topics, "good users", and those that behave unexpectedly, "random users".

The simulations use a data set of 7000 news articles, 500 articles each for fourteen different news categories, such as business, law, and money, collected using the Guardian API.[3]

4.1 Simulated Users

The "random user" refines randomly. For example, **remove document**, deletes a randomly selected document from a randomly selected topic.

Our "good user" reflects a realistic user behavior pattern: identify a mixed category topic and apply refinements to focus the topic on its most dominant category. Thus the "good user"—with access to true document categories—first chooses a topic associated with multiple categories of documents and determines the dominant category of the top documents for the topic. Then, refinement operations push the topic to the dominant category. For example, the "good user" may remove a document which does not belong to the dominant category. Additional simulation are found in Appendix A.

[3]https://open-platform.theguardian.com

4.2 Method

We train forty initial LDA models, twenty with ten topics and twenty with twenty topics for the news articles, resulting in models with less and more topics than the true number of categories.

For each of the three HLTM systems and each of the seven refinement types, we randomly select one of the pre-trained models. The create and split topic refinement types select from the models with ten topics, ensuring that topics have overlapping categories, while the others select from the models with twenty topics. We then apply a refinement as dictated by the simulated user. For the "random user", we randomly select refinement parameters, such as topic and word (Appendix A.1), and for the "good user", we choose topic and refinement parameters intending to improve the topics (Appendix A.2). We apply the refinement (Section 2.3) and run inference until the model converges or reaches a threshold of twenty Gibbs sampling and three EM iterations. We compute control (Section 3) of the refinement and change in topic coherence using NPMI derived from Wikipedia for the top twenty topic words (Lau et al., 2014). We repeat this process 100 times for each refinement type, simulated user, and HLTM system.

5 Informed Priors Listen to Users, while Constraints Produce Coherent Topics

Table 1 shows the per-refinement control and coherence deltas for the three different HLTM systems. As detailed in Appendix B, Kruskal-Wallis tests show that HLTM systems have significantly different ($p < .05$) control scores for all refinements for the "good user" and for all but **remove word** for the "random user." Coherence deltas were also significantly different for all refinements except **add word**, where *const-gibbs* yields consistently higher coherence improvements than the other conditions aside from **remove document**.

For **remove word**, and **merge topics**, all methods provide good control (scores close to 1.0). However, the informed prior methods, *info-vb* and *info-gibbs*, provide more control, for both the random (C_{Rand}) and good (C_{Good}) users, compared to *const-gibbs*. Informed prior methods also excel at refinements that promote topic words,

	const-gibbs			*info-gibbs*			*info-vb*		
	C_{Rand}	C_{Good}	$Q_{Good}{}^*$	C_{Rand}	C_{Good}	$Q_{Good}{}^*$	C_{Rand}	C_{Good}	$Q_{Good}{}^*$
remove w	1.0 (0.0)	1.0 (0.0)	5.4 (9.7)	1.0 (0.0)	1.0 (0.0)	3.0 (8.9)	1.0 (0.0)	1.0, (0.0)	1.2 (5.0)
remove d	1.0 (0.0)	1.0 (0.0)	-1.7 (10.8)	1.0 (0.0)	1.0 (0.0)	.8 (4.5)	.72 (.4)	.85 (.25)	-6.0 (13.2)
merge t	.97 (.05)	1.0 (0.0)	6.3 (8.7)	.96 (.05)	1.0 (0.0)	-.43 (9.3)	.99 (.02)	.99 (.02)	1.4 (9.8)
add w	.82 (.29)	.86 (.24)	3.0 (9.4)	1.0 (0.0)	.98 (.03)	3.1 (6.4)	.98 (.04)	.98 (.02)	1.7 (5.6)
create t	.08 (.10)	.81 (.13)	-6.6 (13.7)	.98 (.11)	.98 (.04)	-11 (10.4)	1.0 (0.0)	1.0 (0.0)	-13.0 (8.4)
split t	.91 (.09)	.79 (.19)	1.9 (17.9)	.93 (.06)	.87 (.19)	-7.9 (13.5)	1.0 (0.0)	.93 (.16)	-1.6 (8)
reorder w	.41 (.53)	.19 (.20)	1.6 (7)	1.19 (.46)	.56 (.24)	-1.0 (5.5)	1.02 (.27)	.44 (.24)	-1.0 (5.1)

Table 1: Simulation results, reported as *mean (SD)*: control with the random (C_{Rand}) and good (C_{Good}) users, and coherence deltas (Q_{Good}) for the good user (we omit coherence for the random user as the goal there is not to improve the topics). *values reported as E-04.

such as **add word** and **create topic**. On the other hand, *const-gibbs* supports defining token and document-level constraints, which ensure almost perfect control for refinements that require restricting certain words or documents, such as **remove word** and **remove document**.

Additionally, comparing good and random users, all systems provide similar control except for *const-gibbs* for **create topic**: .81 for good (C_{Good}) compared to .08 for random (C_{Rand}). This is because *const-gibbs* is limited by the underlying data and cannot generate topics containing random, unrelated seed words, lowering control for the "random user." Informed prior models, however, inflate priors to adhere to user feedback, regardless of whether it aligns with the underlying data, so these methods provide higher control even for random input. Finally, for **change word order**, all three systems lack control. As topic models are probabilistic models, it is therefore difficult to maintain the exact user provided word order.

5.1 Why Informed Priors Offer Control

Informed priors provide higher control than constraints for refinements that require promoting words, such as **add word** and **create topic**. To understand the difference between these two feedback techniques, we conduct an additional simulation to compare *const-gibbs* and *info-gibbs*: we generate an initial topic model of 10 topics and apply **add word** refinements to explore varied control of the feedback techniques.

The initial model includes a law topic with the top ten words: "court, law, justice, rights, legal, case, police, human, public, courts". A user wants to add the word "injustice", initially ranked at 1035th position, to this topic using both *const-gibbs* and *info-gibbs* models. While *const-gibbs* improves the ranking of the added word to 631, *info-gibbs* puts this word at the first position in the updated topic. The *const-gibbs* system tries to push tokens of "injustice" to the *law* topic; however, there just are not enough occurrences to put it in the first ten words. Even assigning all its occurrences to the law topic cannot improve its ranking further. On the other hand, *info-gibbs* can increase the prior for "injustice" enough to put the word in the top of the topic list; until overruled by data *info-gibbs*, can use high priors to incorporate user feedback, resulting in higher control.

6 Conclusion

Informed prior models provide an effective way to incorporate different feedback into topic models, improving *user control* and topic coherence, while constraints yield higher quality topics, but with less control. While we simulate user behavior for good and random users, future work should compare these systems with end users, as well as compare end user ratings of control with our proposed automated metrics.

Interactive models—by design—are balancing user insight with the truth of the data (and thus the world). An important question for future models, especially interactive ones, is how to signal to the user when their desires do not comport with reality. In such cases, control may not be a desired property of interactive systems.

Acknowledgements

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-1822494 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

Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4):105–120.

David Andrzejewski, Xiaojin Zhu, and Mark Craven. 2009. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In *Proceedings of the International Conference of Machine Learning*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Jordan Boyd-Graber, Yuening Hu, and David Mimno. 2017. *Applications of Topic Models*, volume 11 of *Foundations and Trends in Information Retrieval*. NOW Publishers.

Jordan Boyd-Graber, David Mimno, David Newman, Edoardo M Airoldi, David Blei, and Elena A Erosheva. 2014. Care and feeding of topic models: Problems, diagnostics, and improvements. *Handbook of mixed membership models and their applications*, pages 3–34.

Jonathan Chang, Jordan L Boyd-Graber, Sean Gerrish, Chong Wang, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Proceedings of Advances in Neural Information Processing Systems*.

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):1992–2001.

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 2017 CHI Conference on Human Factors in Computing Systems*.

Angela Fan, Finale Doshi-Velez, and Luke Miratrix. 2017. Prior matters: simple and general methods for evaluating and improving topic quality in topic modeling. *arXiv preprint arXiv:1701.03227*.

James Foulds, Shachi Kumar, and Lise Getoor. 2015. Latent topic networks: A versatile probabilistic programming framework for topic models. In *Proceedings of the International Conference of Machine Learning*.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235.

Enamul Hoque and Giuseppe Carenini. 2015. Convisit: Interactive topic modeling for exploring asynchronous online conversations. In *International Conference on Intelligent User Interfaces*. ACM.

Yuening Hu, Jordan Boyd-Graber, Brianna Satinoff, and Alison Smith. 2014. Interactive topic modeling. *Machine learning*, 95(3).

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539.

Tak Yeon Lee, Smith Alison, Kevin Seppi, Niklas Elmqvist, Jordan Boyd-Graber, and Leah Findlater. 2017. The human touch: How non-expert users perceive, interpret, and fix topic models. *International Journal of Human-Computer Studies*.

David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Chris Musialek, Philip Resnik, and S Andrew Stavisky. 2016. Using text analytic techniques to create efficiencies in analyzing qualitative data: A comparison between traditional content analysis and a topic modeling approach. *American Association for Public Opinion Research*.

Quentin Pleplé. 2013. Interactive topic modeling. Master's thesis, UC San Diego.

Margaret E Roberts, Brandon M Stewart, Dustin Tingley, Christopher Lucas, Jetson Leder-Luis, Shana Kushner Gadarian, Bethany Albertson, and David G Rand. 2014. Structural topic models for open-ended survey responses. *American Journal of Political Science*, 58(4):1064–1082.

Ben Shneiderman. 2010. *Designing the user interface: strategies for effective human-computer interaction*. Pearson Education India.

Alison Smith, Varun Kumar, Jordan Boyd-Graber, Kevin Seppi, and Leah Findlater. 2018. Closing the loop: User-centered design and evaluation of a human-in-the-loop topic modeling system. In *International Conference on Intelligent User Interfaces*. ACM.

Jun Wang, Changsheng Zhao, Junfu Xiang, and Kanji Uchino. 2019. Interactive topic model with enhanced interpretability. In *IUI Workshops*.

Pengtao Xie, Diyi Yang, and Eric P Xing. 2015. Incorporating word correlation knowledge into topic modeling. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

Yi Yang, Doug Downey, Jordan L Boyd-Graber, and Jordan Boyd Graber. 2015. Efficient methods for incorporating knowledge into topic models. In *Proceedings of Empirical Methods in Natural Language Processing*.

Ke Zhai, Jordan L. Boyd-Graber, Nima Asadi, and Mohamad L. Alkhouja. 2012. Mr. LDA: a flexible large scale topic modeling package using variational inference in mapreduce. In *Proceedings of the 21st International Conference on World Wide Web*.

A Simulation Details

To simulate the behavior of the "random user" and "good user" for the three HLTM systems, we train 40 initial LDA models, 20 with 10 topics and 20 with 20 topics for the news articles, resulting in models with less and more topics than the true number of categories.

A.1 Random User Simulation

To simulate random user behavior, for each of the three systems and for each of the seven refinement types, we randomly select a pre-trained LDA model from the pool of models with 20 topics. Then, we apply a refinement of that refinement type to the selected model. We randomly select refinement specific parameters, such as candidate topic, word to be added, and document to be deleted. We run inference until the model converges or reaches a limit. For Gibbs sampling models, *info-gibbs* and *const-gibbs*, we use 20 iterations as limit and for the variational model, *info-vb*, we use 3 EM iterations as the limit. After applying the refinement, we compute the *control* and coherence given the updated and initial model. We perform this 100 times for each of the refinement types and HLTM systems.

A.2 Good User Simulation

For each category c of the 14 categories of the Guardian news dataset (art & design, business, education, environment, fashion, film, football, law, money, music, politics, science, sports, technology), we compute the most important words in c, S_c, using a Logistic regression classifier. We use S_c as a list of representative words for category c.

Given a labeled corpus, we randomly choose one of the pre-trained models. When applying create or split topic refinement types, we select from the models with 10 topics, ensuring that topics have overlapping categories. While applying all other refinement types, we select from the models with 20 topics. We then simulate good user behavior for each of the refinement types as follows:

1. Add word: Randomly select a topic t from those where the top 20 documents are from more than one category. Then, find the corresponding labeled category c by analyzing top 20 documents in the selected category. To improve the topic coherence of t, add top ranked words (from one to five words) from S_c, which are not already in the top words of t.

2. Remove word: Randomly select a topic t from those where top 20 documents are from more than one category. Then, find the corresponding labeled category c by analyzing top 20 documents in the selected category. For selected topic t, remove words which are not part of S_c.

3. Change word order: Randomly select a topic t among all topics. Then, find the corresponding labeled category c by analyzing top 20 documents in the selected category. Then, find words between index 10 to 20, which are at higher rank in S_c. Promote such words to a higher rank using change word order.

4. Remove document: Randomly select a topic t from those where top 20 documents are from more than one category. Then, find the corresponding labeled category c by analyzing top 20 documents in the selected category. For selected topic t, delete documents (from one to five documents), which are not in c.

5. Merge topics: Randomly choose a topic pair to merge which represents a common category c.

6. Create topic: Randomly select a category c which is not a dominant category in any of the topics. Create a topic by providing top 10 words as seed words from S_c.

7. Split topic: Randomly select a topic from those which have documents from two different categories, c_1 and c_2. Split the top 20 words in that topic into two lists using the representative words from S_{c_1} and S_{c_2}. Then, split the topic using one of the lists.

B Kruskal Wallis Tests

We provide details on the Kruskal Wallis tests used to assess whether there are significant differences in how the three HLTM systems, *const-gibbs*, *info-gibbs*, and *info-vb*, impact control and topic coherence. The means reported here repeat what is provided in the main paper, but with the additional χ^2 and p values output from the Kruskal Wallis tests; $p < .05$ is considered to be significant.

Because control values are not comparable across the seven *user-preferred* refinements, we conducted separate Kruskal Wallis tests for each refinement. The results include control for the simulated good user (Table 3) and for the simulated random user (Table 2), as well as quality improvements (coherence) for the simulated good user (Table 4).

	const-gibbs	*info-gibbs*	*info-vb*	χ^2	p-value
add w	0.82	1.00	0.99	249.35	$< .001$
remove w	1.00	1.00	1.00	0.42	.810
remove d	1.00	1.00	0.72	27.12	$< .001$
merge t	0.97	0.96	0.99	31.24	$< .001$
reorder w	0.41	1.19	1.03	113.52	$< .001$
create t	0.08	0.98	1.00	277.23	$< .001$
split t	0.91	0.93	1.00	119.47	$< .001$

Table 2: Average control provided by the three HLTM systems for seven *user-preferred* refinements and simulated random user behavior. Kruskal-Wallis tests ($p < .05$) show significant differences between the systems for all refinements except remove word.

	const-gibbs	*info-gibbs*	*info-vb*	χ^2	p-value
add w	0.86	0.98	0.98	13.02	.001
remove w	0.99	0.99	0.99	6.22	.045
remove d	0.99	0.99	0.85	163.73	$< .001$
merge t	1.00	1.00	0.99	22.76	$< .001$
reorder w	0.19	0.56	0.44	103.44	$< .001$
create t	0.82	0.98	1.00	191.82	$< .001$
split t	0.77	0.87	0.93	81.71	$< .001$

Table 3: Average control provided by the three HLTM systems for seven *user-preferred* refinements and simulated good user behavior. Kruskal-Wallis tests ($p < .05$) show significant differences between the systems for all refinements.

	const-gibbs	*info-gibbs*	*info-vb*	χ^2	p-value
add w	3.0E-04	3.1E-04	1.7E-04	2.93	.230
remove w	5.3E-04	3.0E-04	1.2E-04	25.51	$< .001$
remove d	-1.7E-04	7.5E-05	-6.0E-04	19.29	$< .001$
merge t	6.3E-04	-4.3E-05	1.4E-04	30.66	$< .001$
reorder w	1.6E-04	-8.0E-05	-1.0E-05	7.67	.020
create t	-6.6E-04	-1.1E-03	-1.2E-03	11.20	.004
split t	1.9E-04	-7.9E-04	-1.6E-04	22.19	$< .001$

Table 4: Average coherence provided by the three HLTM systems for seven *user-preferred* refinements and simulated good user behavior. Kruskal-Wallis tests ($p < .05$) show significant differences between the systems for all refinements except for add word.