Yuening Hu, **Jordan Boyd-Graber**, and Brianna Satinoff. **Interactive Topic Modeling**. *Association for Computational Linguistics*, 2011.

```
    @inproceedings{Hu:Boyd-Graber:Satinoff-2011,
Title = {Interactive Topic Modeling},
Booktitle = {Association for Computational Linguistics},
Author = {Yuening Hu and Jordan Boyd-Graber and Brianna Satinoff},
Year = {2011},
Location = {Portland, Oregon},
}
```

Interactive Topic Modeling

Yuening Hu
Department of Computer Science
University of Maryland
ynhu@cs.umd.edu

Jordan Boyd-Graber
iSchool
University of Maryland
jbg@umiacs.umd.edu

Brianna Satinoff
Department of Computer Science
University of Maryland
bsonrisa@cs.umd.edu

Abstract

Topic models have been used extensively as a tool for corpus exploration, and a cottage industry has developed to tweak topic models to better encode human intuitions or to better model data. However, creating such extensions requires expertise in machine learning unavailable to potential end-users of topic modeling software. In this work, we develop a framework for allowing users to iteratively refine the topics discovered by models such as latent Dirichlet allocation (LDA) by adding constraints that enforce that sets of words must appear together in the same topic. We incorporate these constraints interactively by selectively removing elements in the state of a Markov Chain used for inference; we investigate a variety of methods for incorporating this information and demonstrate that these interactively added constraints improve topic usefulness for simulated and actual user sessions.

1 Introduction

Probabilistic topic models, as exemplified by probabilistic latent semantic indexing (Hofmann, 1999) and latent Dirichlet allocation (LDA) (Blei et al., 2003) are unsupervised statistical techniques to discover the thematic topics that permeate a large corpus of text documents. Topic models have had considerable application beyond natural language processing in computer vision (Rob et al., 2005), biology (Shringarpure and Xing, 2008), and psychology (Landauer et al., 2006) in addition to their canonical application to text.

For text, one of the few real-world applications of topic models is corpus exploration. Unannotated, noisy, and ever-growing corpora are the norm rather than the exception, and topic models offer a way to quickly get the gist a large corpus.[1]

Contrary to the impression given by the tables shown in topic modeling papers, topics discovered by topic modeling don't always make sense to ostensible end users. Part of the problem is that the objective function of topic models doesn't always correlate with human judgements (Chang et al., 2009). Another issue is that topic models — with their bag-of-words vision of the world — simply lack the necessary information to create the topics as end-users expect.

There has been a thriving cottage industry adding more and more information to topic models to correct these shortcomings; either by modeling perspective (Paul and Girju, 2010; Lin et al., 2006), syntax (Wallach, 2006; Gruber et al., 2007), or authorship (Rosen-Zvi et al., 2004; Dietz et al., 2007). Similarly, there has been an effort to inject human knowledge into topic models (Boyd-Graber et al., 2007; Andrzejewski et al., 2009; Petterson et al., 2010).

However, these are *a priori* fixes. They don't help a frustrated **consumer** of topic models staring at a collection of topics that don't make sense. In this paper, we propose interactive topic modeling (ITM), an *in situ* method for incorporating human knowledge into topic models. In Section 2, we review prior work on creating probabilistic models that incorporate human knowledge, which we extend in Section 3 to apply to ITM sessions. Section 4 discusses the implementation of this process during the inference process. Via a motivating example in Section 5, simulated ITM sessions in Section 6, and a real interactive test in Section 7, we demonstrate that our approach is able to focus a user's desires in a topic model, better capture the key properties of a corpus, and capture diverse interests from users on the web.

[1] For examples, see Rexa http://rexa.info/, JSTOR http://showcase.jstor.org/blei/, and the NIH https://app.nihmaps.org/nih/.

2 Putting Knowledge in Topic Models

At a high level, topic models such as LDA take as input a number of topics K and a corpus. As output, a topic model discovers K distributions over words — the namesake topics — and associations between documents and topics. In LDA both of these outputs are multinomial distributions; typically they are presented to users in summary form by listing the elements with highest probability. For an example of topics discovered from a 20-topic model of New York Times editorials, see Table 1.

When presented with poor topics learned from data, users can offer a number of complaints:[2] these documents should have similar topics but don't (Daumé III, 2009); this topic should have syntactic coherence (Gruber et al., 2007; Boyd-Graber and Blei, 2008); this topic doesn't make any sense at all (Newman et al., 2010); this topic shouldn't be associated with this document but is (Ramage et al., 2009); these words shouldn't be the in same topic but are (Andrzejewski et al., 2009); or these words should be in the same topic but aren't (Andrzejewski et al., 2009).

Many of these complaints can be addressed by using "must-link" constraints on topics, retaining Andrzejewski et al's (2009) terminology borrowed from the database literature. A "must-link" constraint is a group of words whose probability must be correlated in the topic. For example, Figure 1 shows an example constraint: {plant, factory}. After this constraint is added, the probabilities of "plant" and "factory" in each topic are likely to both be high or both be low. It's unlikely for "plant" to have high probability in a topic and "factory" to have a low probability. In the next section, we demonstrate how such constraints can be built into a model and how they can even be added while inference is underway.

In this paper, we view constraints as transitive; if "plant" is in a constraint with "factory" and "factory" is in a constraint with "production," then "plant" is in a constraint with "production." Making this assumption can simplify inference slightly, which we take advantage of in Section 3.1, but the real reason for this assumption is because not doing so would introduce ambiguity over the path associated with an observed token in the generative process. As long as a word is either in a single constraint or in the general vocabulary, there is only a single path. The details of this issue are further discussed in Section 4.

[2]Citations in this litany of complaints are offline solutions for addressing the problem; the papers also give motivation why such complaints might arise.

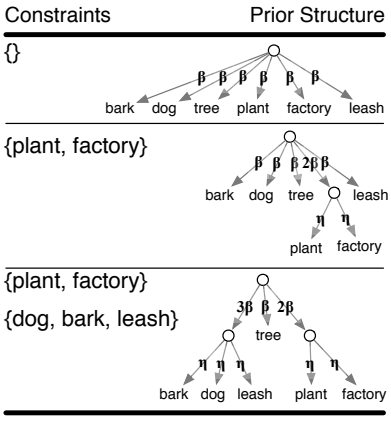

Figure 1: How adding constraints (left) creates new topic priors (right). The trees represent correlated distributions (assuming $\eta >> \beta$). After the {plant, factory} constraint is added, it is now highly unlikely for a topic drawn from the distribution to have a high probability for "plant" and a low probability for "factory" or vice versa. The bottom panel adds an additional constraint, so now dog-related words are also correlated. Notice that the two constraints themselves are uncorrelated. It's possible for both, either, or none of "bark" and "plant" (for instance) to have high probability in a topic.

3 Constraints Shape Topics

As discussed above, LDA views topics as distributions over words, and each document expresses an admixture of these topics. For "vanilla" LDA (no constraints), these are symmetric Dirichlet distributions. A document is composed of a number of observed words, which we call tokens to distinguish specific observations from the more abstract word (type) associated with each token. Because LDA assumes a document's tokens are interchangeable, it treats the document as a bag-of-words, ignoring potential relations between words.

This problem with vanilla LDA can be solved by encoding constraints, which will "guide" different words into the same topic. Constraints can be added to vanilla LDA by replacing the multinomial distribution over words for each topic with a collection of

tree-structured multinomial distributions drawn from a prior as depicted in Figure 1. By encoding word distributions as a tree, we can preserve conjugacy and relatively simple inference while encouraging correlations between related concepts (Boyd-Graber et al., 2007; Andrzejewski et al., 2009; Boyd-Graber and Resnik, 2010). Each topic has a top-level distribution over words and constraints, and each constraint in each topic has second-level distribution over the words in the constraint. Critically, the per-constraint distribution over words is engineered to be non-sparse and close to uniform. The top level distribution encodes which constraints (and unconstrained words) to include; the lower-level distribution forces the probabilities to be correlated for each of the constraints.

In LDA, a document's token is produced in the generative process by choosing a topic z and sampling a word from the multinomial distribution ϕ_z of topic z. For a constrained topic, the process now can take two steps. First, a first-level node in the tree is selected from ϕ_z. If that is an unconstrained word, the word is emitted and the generative process for that token is done. Otherwise, if the first level node is constraint l, then choose a word to emit from the constraint's distribution over words $\pi_{z,l}$.

More concretely, suppose for a corpus with M documents we have a set of constraints $\mathbf{\Omega}$. The prior structure has B branches (one branch for each word not in a constraint and one for each constraint). Then the generative process for constrained LDA is:

1. For each topic $i \in \{1, \dots K\}$:
   (a) draw a distribution over the B branches (words and constraints) $\phi_i \sim \text{Dir}(\vec{\beta})$, and
   (b) for each constraint $\Omega_j \in \mathbf{\Omega}$, draw a distribution over the words in the constraint $\pi_{i,j} \sim \text{Dir}(\eta)$, where $\pi_{i,j}$ is a distribution over the words in Ω_j
2. Then for each document $d \in \{1, \dots M\}$:
   (a) first draw a distribution over topics $\theta_d \sim \text{Dir}(\alpha)$,
   (b) then for each token $n \in \{1, \dots N_d\}$:
      i. choose a topic assignment $z_{d,n} \sim \text{Mult}(\theta_d)$, and then
      ii. choose either a constraint or word from $\text{Mult}(\phi_{z_{d,n}})$:
         A. if we chose a word, emit that word $w_{d,n}$
         B. otherwise if we chose a constraint index $l_{d,n}$, emit a word $w_{d,n}$ from the constraint's distribution over words in topic $z_{d,n}$: $w_{d,n} \sim \text{Mult}(\pi_{z_{d,n},l_{d,n}})$.

In this model, α, β, and η are Dirichlet hyperparameters set by the user; their role is explained below.

3.1 Gibbs Sampling for Topic Models

In topic modeling, collapsed Gibbs sampling (Griffiths and Steyvers, 2004) is a standard procedure for obtaining a Markov chain over the latent variables in the model. Given certain technical conditions, the stationary distribution of the Markov chain is the posterior (Neal, 1993). Given M documents the state of a Gibbs sampler for LDA consists of topic assignments for each token in the corpus and is represented as $Z = \{z_{1,1} \dots z_{1,N_1}, z_{2,1}, \dots z_{M,N_M}\}$. In each iteration, every token's topic assignment $z_{d,n}$ is resampled based on topic assignments for all the tokens except for $z_{d,n}$. (This subset of the state is denoted $\boldsymbol{Z}_{-(d,n)}$). The sampling equation for $z_{d,n}$ is

$$p(z_{d,n} = k|\mathbf{Z}_{-(d,n)}, \alpha, \beta) \propto \frac{T_{d,k} + \alpha}{T_{d,\cdot} + K\alpha} \frac{P_{k,w_{d,n}} + \beta}{P_{k,\cdot} + V\beta} \quad (1)$$

where $T_{d,k}$ is the number of times topic k is used in document d, $P_{k,w_{d,n}}$ is the number of times the type $w_{d,n}$ is assigned to topic k, and α, β are the hyperparameters of the two Dirichlet distributions, and B is the number of top-level branches (this is the vocabulary size for vanilla LDA). When a dot replaces a subscript of a count, it represents the marginal sum over all possible topics or words, e.g. $T_{d,\cdot} = \sum_k T_{d,k}$. The count statistics P and T provide summaries of the state. Typically, these only change based on assignments of latent variables in the sampler; in Section 4 we describe how changes in the model's *structure* (in addition to the latent state) can be reflected in these count statistics.

Contrasting with the above inference is the inference for a constrained model. (For a derivation, see Boyd-Graber, Blei, and Zhu (2007) for the general case or Andrzejewski, Zhu, and Craven (2009) for the specific case of constraints.) In this case the sampling equation for $z_{d,n}$ is changed to $p(z_{d,n} = k|\mathbf{Z}_{-(d,n)}, \alpha, \beta, \eta)$

$$\propto \begin{cases} \frac{T_{d,k}+\alpha}{T_{d,\cdot}+K\alpha} \frac{P_{k,w_{d,n}}+\beta}{P_{k,\cdot}+V\beta} & \text{if } \forall l, w_{d,n} \notin \Omega_l \\ \frac{T_{d,k}+\alpha}{T_{d,\cdot}+K\alpha} \frac{P_{k,l}+C_l\beta}{P_{k,\cdot}+V\beta} \frac{W_{k,l,w_{d,n}}+\eta}{W_{k,l,\cdot}+C_l\eta} & w_{d,n} \in \Omega_l \end{cases}, \quad (2)$$

where $P_{k,w_{d,n}}$ is the number of times the unconstrained word $w_{d,n}$ appears in topic k; $P_{k,l}$ is the

number of times any word of constraint Ω_l appears in topic k; $W_{k,l,w_{d,n}}$ is the number of times word $w_{d,n}$ appears in constraint Ω_l in topic k; V is the vocabulary size; C_l is the number of words in constraint Ω_l. Note the differences between these two samplers for constrained words; however, for unconstrained LDA and for unconstrained words in constrained LDA, the conditional probability is the same.

In order to make the constraints effective, we set the constraint word-distribution hyperparameter η to be much larger than the hyperparameter for the distribution over constraints and vocabulary β. This gives the constraints higher weight. Normally, estimating hyperparameters is important for topic modeling (Wallach et al., 2009). However, in ITM, sampling hyperparameters often (but not always) undoes the constraints (by making η comparable to β), so we keep the hyperparameters fixed.

4 Interactively adding constraints

For a static model, inference in ITM is the same as in previous models (Andrzejewski et al., 2009). In this section, we detail how interactively changing constraints can be accommodated in ITM, smoothly transitioning from unconstrained LDA (n.b. Equation 1) to constrained LDA (n.b. Equation 2) with one constraint, to constrained LDA with two constraints, etc.

A central tool that we will use is the strategic unassignment of states, which we call ablation (distinct from feature ablation in supervised learning). As described in the previous section, a sampler stores the topic assignment of each token. In the implementation of a Gibbs sampler, unassignment is done by setting a token's topic assignment to an invalid topic (e.g. -1, as we use here) and decrementing any counts associated with that word.

The constraints created by users implicitly signal that words in constraints don't belong in a given topic. In other models, this input is sometimes used to "fix," i.e. deterministically hold constant topic assignments (Ramage et al., 2009). Instead, we change the underlying model, using the current topic assignments as a starting position for a new Markov chain with some states strategically unassigned. How much of the existing topic assignments we use leads to four different options, which are illustrated in Figure 2.

	Previous	New
All	[bark:2, dog:3, leash:3 dog:2] [bark:2, bark:2, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]	[bark:-1, dog:-1, leash:-1 dog:-1] [bark:-1, bark:-1, plant:-1, tree:-1] [tree:-1,play:-1,forest:-1,leash:-1]
Doc	[bark:2, dog:3, leash:3 dog:2] [bark:2, bark:2, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]	[bark:-1, dog:-1, leash:-1 dog:-1] [bark:-1, bark:-1, plant:-1, tree:-1] [tree:2,play:2,forest:1,leash:2]
Term	[bark:2, dog:3, leash:3 dog:3] [bark:2, bark:2, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]	[bark:-1, dog:-1, leash:3 dog:-1] [bark:-1, bark:-1, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]
None	[bark:2, dog:3, leash:3 dog:2] [bark:2, bark:2, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]	[bark:2, dog:3, leash:3 dog:2] [bark:2, bark:2, plant:2, tree:3] [tree:2,play:2,forest:1,leash:2]

Figure 2: Four different strategies for state ablation after the words "dog" and "bark" are added to the constraint {"leash," "puppy"} to make the constraint {"dog," "bark," "leash," "puppy"}. The state is represented by showing the current topic assignment after each word (e.g. "leash" in the first document has topic 3, while "forest" in the third document has topic 1). On the left are the assignments before words were added to constraints, and on the right are the ablated assignments. Unassigned words are given the new topic assignment -1 and are highlighted in red.

All We could revoke all state assignments, essentially starting the sampler from scratch. This does not allow *interactive* refinement, as there is nothing to enforce that the new topics will be in any way consistent with the existing topics. Once the topic assignments of all states are revoked, the counts for T, P and W (as described in Section 3.1) will be zero, retaining no information about the state the user observed.

Doc Because topic models treat the document context as exchangeable, a document is a natural context for partial state ablation. Thus if a user adds a set of words $\boldsymbol{S}$ to constraints, then we have reason to suspect that all documents containing any one of $\boldsymbol{S}$ may have incorrect topic assignments. This is reflected in the state of the sampler by performing the UNASSIGN (Algorithm 1) operation for each word in any document containing a word added to a constraint.

Algorithm 1 UNASSIGN$(d, n, w_{d,n}, z_{d,n} = k)$

1: $T:\ T_{d,k} \leftarrow T_{d,k} - 1$
2: If $w_{d,n} \notin \Omega^{old}$,
  $P:\ P_{k,w_{d,n}} \leftarrow P_{k,w_{d,n}} - 1$
3: Else: suppose $w_{d,n} \in \Omega^{old}_m$,
  $P:\ P_{k,m} \leftarrow P_{k,m} - 1$
  $W:\ W_{k,m,w_{d,n}} \leftarrow W_{k,m,w_{d,n}} - 1$

This is equivalent to the Gibbs2 sampler of Yao et al. (2009) for incorporating new documents in a streaming context. Viewed in this light, a user is using words to select documents that should be treated as "new" for this refined model.

Term Another option is to perform ablation only on the topic assignments of tokens whose words have added to a constraint. This applies the unassignment operation (Algorithm 1) only to tokens whose corresponding word appears in added constraints (i.e. a subset of the Doc strategy). This makes it less likely that other tokens in similar contexts will follow the words explicitly included in the constraints to new topic assignments.

None The final option is to move words into constraints but keep the topic assignments fixed. Thus, P and W change, but not T, as described in Algorithm 2.[3] This is arguably the simplest option, and in principle is sufficient, as the Markov chain should find a stationary distribution regardless of the starting position. In practice, however, this strategy is less interactive, as users don't feel that their constraints are actually incorporated in the model, and inertia can keep the chain from reflecting the constraints.

Algorithm 2 MOVE$(d, n, w_{d,n}, z_{d,n} = k, \Omega_l)$

1: If $w_{d,n} \notin \Omega^{old}$,
$P:\ P_{k,w_{d,n}} \leftarrow P_{k,w_{d,n}} - 1,\ \ P_{k,l} \leftarrow P_{k,l} + 1$
$W:\ W_{k,l,w_{d,n}} \leftarrow W_{k,l,w_{d,n}} + 1$
2: Else, suppose $w_{d,n} \in \Omega_m^{old}$,
$P:\ P_{k,m} \leftarrow P_{k,m} - 1,\ P_{k,l} \leftarrow P_{k,l} + 1$
$W:\ W_{k,m,w_{d,n}} \leftarrow W_{k,m,w_{d,n}} - 1$
$W_{k,l,w_{d,n}} \leftarrow W_{k,l,w_{d,n}} + 1$

Regardless of what ablation scheme is used, after the state of the Markov chain is altered, the next step is to actually run inference forward, sampling assignments for the unassigned tokens for the "first" time and changing the topic assignment of previously assigned tokens. How many additional iterations are required after adding constraints is a delicate tradeoff between interactivity and effectiveness, which we investigate further in the next sections.

[3]This assumes that there is only one possible path in the constraint tree that can generate a word; in other words, this assumes that constraints are transitive, as discussed at the end of Section 2. In the more general case, when words lack a unique path in the constraint tree, an additional latent variable specifies which possible paths in the constraint tree produced the word; this would have to be sampled. All other updating strategies are immune to this complication, as the assignments are left unassigned.

5 Motivating Example

To examine the viability of ITM, we begin with a qualitative demonstration that shows the potential usefulness of ITM. For this task, we used a corpus of about 2000 New York Times editorials from the years 1987 to 1996. We started by finding 20 initial topics with no constraints, as shown in Table 1 (left).

Notice that topics 1 and 20 both deal with Russia. Topic 20 seems to be about the Soviet Union, with topic 1 about the post-Soviet years. We wanted to combine the two into a single topic, so we created a constraint with all of the clearly Russian or Soviet words (*boris, communist, gorbachev, mikhail, russia, russian, soviet, union, yeltsin*). Running inference forward 100 iterations with the **Doc** ablation strategy yields the topics in Table 1 (right). The two Russia topics were combined into Topic 20. This combination also pulled in other relevant words that not near the top of either topic before: "moscow" and "relations." Topic 1 is now more about elections in countries other than Russia. The other 18 topics changed little.

While we combined the Russian topics, other researchers analyzing large corpora might preserve the Soviet vs. post-Soviet distinction but combine topics about American government. ITM allows tuning for specific tasks.

6 Simulation Experiment

Next, we consider a process for evaluating our ITM using automatically derived constraints. These constraints are meant to simulate a user with a predefined list of categories (e.g. reviewers for journal submissions, e-mail folders, etc.). The categories grow more and more specific during the session as the simulated users add more constraint words.

To test the ability of ITM to discover relevant subdivisions in a corpus, we use a dataset with predefined, intrinsic labels and assess how well the discovered latent topic structure can reproduce the corpus's inherent structure. Specifically, for a corpus with M classes, we use the per-document topic distribution as a feature vector in a supervised classi-

Topic	Words
1	election, yeltsin, russian, political, party, democratic, russia, president, democracy, boris, country, south, years, month, government, vote, since, leader, presidential, military
2	new, york, city, state, mayor, budget, giuliani, council, cuomo, gov, plan, year, rudolph, dinkins, lead, need, governor, legislature, pataki, david
3	nuclear, arms, weapon, defense, treaty, missile, world, unite, yet, soviet, lead, secretary, would, control, korea, intelligence, test, nation, country, testing
4	president, bush, administration, clinton, american, force, reagan, war, unite, lead, economic, iraq, congress, america, iraqi, policy, aid, international, military, see
⋮	⋮
20	soviet, lead, gorbachev, union, west, mikhail, reform, change, europe, leaders, poland, communist, know, old, right, human, washington, western, bring, party

Topic	Words
1	election, democratic, south, country, president, party, africa, lead, even, democracy, leader, presidential, week, politics, minister, percent, voter, last, month, years
2	new, york, city, state, mayor, budget, council, giuliani, gov, cuomo, year, rudolph, dinkins, legislature, plan, david, governor, pataki, need, cut
3	nuclear, arms, weapon, treaty, defense, war, missile, may, come, test, american, world, would, need, lead, get, join, yet, clinton, nation
4	president, administration, bush, clinton, war, unite, force, reagan, american, america, make, nation, military, iraq, iraqi, troops, international, country, yesterday, plan
⋮	⋮
20	soviet, union, economic, reform, yeltsin, russian, lead, russia, gorbachev, leaders, west, president, boris, moscow, europe, poland, mikhail, communist, power, relations

Table 1: Five topics from a 20 topic topic model on the editorials from the New York times before adding a constraint (left) and after (right). After the constraint was added, which encouraged Russian and Soviet terms to be in the same topic, non-Russian terms gained increased prominence in Topic 1, and "Moscow" (which was not part of the constraint) appeared in Topic 20.

fier (Hall et al., 2009). The lower the classification error rate, the better the model has captured the structure of the corpus.[4]

6.1 Generating automatic constraints

We used the 20 Newsgroups corpus, which contains 18846 documents divided into 20 constituent newsgroups. We use these newsgroups as ground-truth labels.[5]

We simulate a user's constraints by ranking words in the training split by their information gain (IG).[6] After ranking the top 200 words for each class by IG, we delete words associated with multiple labels to prevent constraints for different labels from merging. The smallest class had 21 words remaining after removing duplicates (due to high overlaps of 125 words between "talk.religion.misc" and "soc.religion.christian," and 110 words between "talk.religion.misc" and "alt.atheism"), so the top 21 words for each class were the ingredients for our simulated constraints. For example, for the class "soc.religion.christian," the 21 constraint words include "catholic, scripture, resurrection, pope, sabbath, spiritual, pray, divine, doctrine, orthodox." We simulate a user's ITM session by adding a word to each of the 20 constraints until each of the constraints has 21 words.

6.2 Simulation scheme

Starting with 100 base iterations, we perform successive rounds of refinement. In each round a new constraint is added corresponding to the newsgroup labels. Next, we perform one of the strategies for state ablation, add additional iterations of Gibbs sampling, use the newly obtained topic distribution of each document as the feature vector, and perform classification on the test / train split. We do this for 21 rounds until each label has 21 constraint words. The number of LDA topics is set to 20 to match the number of newsgroups. The hyperparameters for all experiments are $\alpha = 0.1, \beta = 0.01$, and $\eta = 100$.

At 100 iterations, the chain is clearly not converged. However, we chose this number of iterations because it more closely matches the likely use case as users do not wait for convergence. Moreover, while investigations showed that the patterns shown in Fig-

[4] Our goal is to understand the phenomena of ITM, not classification, so these classification results are well below state of the art. However, adding interactively selected topics to the state of the art features (tf-idf unigrams) gives a relative error reduction of 5.1%, while just adding topics from vanilla LDA gives a relative error reduction of 1.1%. Both measurements were obtained without tuning or weighting features, so presumably better results are possible.

[5] http://people.csail.mit.edu/jrennie/20Newsgroups/ In preprocessing, we deleted short documents, leaving 15160 documents, including 9131 training documents and 6029 test documents (default split). Tokenization, lemmatization, and stopword removal was performed using the Natural Language Toolkit (Loper and Bird, 2002). Topic modeling was performed using the most frequent 5000 lemmas as the vocabulary.

[6] IG is computed by the Rainbow toolbox http://www.cs.umass.edu/ mccallum/bow/rainbow/

ure 4 were broadly consistent with larger numbers of iterations, such configurations sometimes had too much inertia to escape from local extrema. More iterations make it harder for the constraints to influence the topic assignment.

6.3 Investigating Ablation Strategies

First, we investigate which ablation strategy best allows constraints to be incorporated. Figure 3 shows the classification error of six different ablation strategies based on the number of words in each constraint, ranging from 0 to 21. Each is averaged over five different chains using 10 additional iterations of Gibbs sampling per round (other numbers of iterations are discussed in Section 6.4). The model runs forward 10 iterations after the first round, another 10 iterations after the second round, etc. In general, as the number of words per constraint increases, the error decreases as models gain more information about the classes.

Strategy **Null** is the non-interactive baseline that contains no constraints (vanilla LDA), but runs inference for a comparable number of rounds. **All Initial** and **All Full** are non-interactive baselines with all constraints known *a priori*. **All Initial** runs the model for the only the initial number of iterations (100 iterations in this experiment), while **All Full** runs the model for the total number of iterations added for the interactive version. (That is, if there were 21 rounds and each round of interactive modeling added 10 iterations, **All Full** would have 210 iterations more than **All Initial**).

While **Null** sees no constraints, it serves as an upper baseline for the error rate (lower error being better) but shows the effect of additional inference. **All Full** is a lower baseline for the error rate since it both sees the constraints at the beginning and also runs for the maximum number of total iterations. **All Initial** sees the constraints before the other ablation techniques but it has fewer total iterations.

The **Null** strategy does not perform as well as the interactive versions, especially with larger constraints. Both **All Initial** and **All Full**, however, show a larger variance (as denoted by error bands around the average trends) than the interactive schemes. This can be viewed as akin to simulated annealing, as the interactive search has more freedom to explore in early rounds. As more constraint words are added each round, the model is less free to explore.

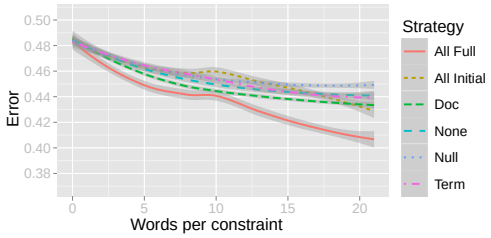

Figure 3: Error rate (y-axis, lower is better) using different ablation strategies as additional constraints are added (x-axis). **Null** represents standard LDA, as the unconstrained baseline. **All Initial** and **All Full** are non-interactive, constrained baselines. The results of **None**, **Term**, **Doc** are more stable (as denoted by the error bars), and the error rate is reduced gradually as more constraint words are added.

The error rate of each interactive ablation strategy is (as expected) between the lower and upper baselines. Generally, the constraints will influence not only the topics of the constraint words, but also the topics of the constraint words' context in the same document. **Doc** ablation gives more freedom for the constraints to overcome the inertia of the old topic distribution and move towards a new one influenced by the constraints.

6.4 How many iterations do users have to wait?

Figure 4 shows the effect of using different numbers of Gibbs sampling iterations after changing a constraint. For each of the ablation strategies, we run $\{10, 20, 30, 50, 100\}$ additional Gibbs sampling iterations. As expected, more iterations reduce error, although improvements diminish beyond 100 iterations. With more constraints, the impact of additional iterations is lessened, as the model has more *a priori* knowledge to draw upon.

For all numbers of additional iterations, while the **Null** serves as the upper baseline on the error rate in all cases, the **Doc** ablation clearly outperforms the other ablation schemes, consistently yielding a lower error rate. Thus, there is a benefit when the model has a chance to relearn the document context when constraints are added. The difference is even larger with more iterations, suggesting **Doc** needs more iterations to "recover" from unassignment.

The luxury of having hundreds or thousands of additional iterations for each constraint would be im-

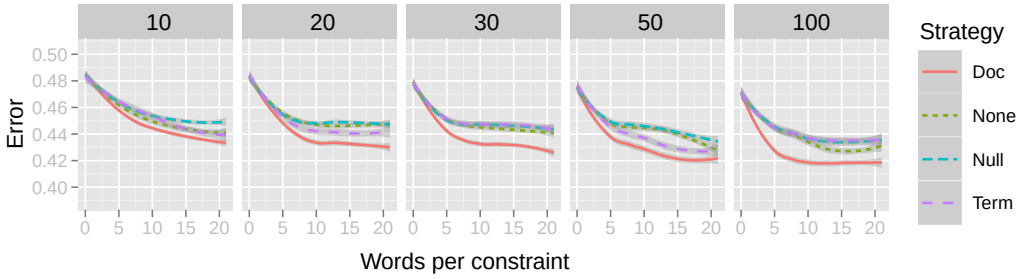

Figure 4: Classification accuracy by strategy and number of additional iterations. The **Doc** ablation strategy performs best, suggesting that the document context is important for ablation constraints. While more iterations are better, there is a tradeoff with interactivity.

practical. For even moderately sized datasets, even one iteration per second can tax the patience of individuals who want to use the system interactively. Based on these results and an *ad hoc* qualitative examination of the resulting topics, we found that 30 additional iterations of inference was acceptable; this is used in later experiments.

7 Getting Humans in the Loop

To move beyond using simulated users adding the same words regardless of what topics were discovered by the model, we needed to expose the model to human users. We solicited approximately 200 judgments from Mechanical Turk, a popular crowdsourcing platform that has been used to gather linguistic annotations (Snow et al., 2008), measure topic quality (Chang et al., 2009), and supplement traditional inference techniques for topic models (Chang, 2010). After presenting our interface for collecting judgments, we examine the results from these ITM sessions both quantitatively and qualitatively.

7.1 Interface for soliciting refinements

Figure 5 shows the interface used in the Mechanical Turk tests. The left side of the screen shows the current topics in a scrollable list, with the top 30 words displayed for each topic.

Users create constraints by clicking on words from the topic word lists. The word lists use a color-coding scheme to help the users keep track of which words they are currently grouping into constraints. The right side of the screen displays the existing constraints. Users can click on icons to edit or delete each one. The constraint currently being built is also shown.

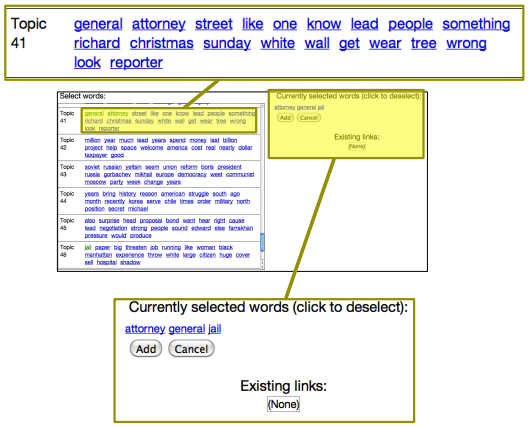

Figure 5: Interface for Mechanical Turk experiments. Users see the topics discovered by the model and select words (by clicking on them) to build constraints to be added to the model.

Clicking on a word will remove that word from the current constraint.

As in Section 6, we can compute the classification error for these users as they add words to constraints. The best users, who seemed to understand the task well, were able to decrease classification error. (Figure 6). The median user, however, had an error reduction indistinguishable from zero. Despite this, we can examine the users' behavior to better understand their goals and how they interact with the system.

7.2 Untrained users and ITM

Most of the large (10+ word) user-created constraints corresponded to the themes of the individual newsgroups, which users were able to infer from the discovered topics. Common constraint themes that

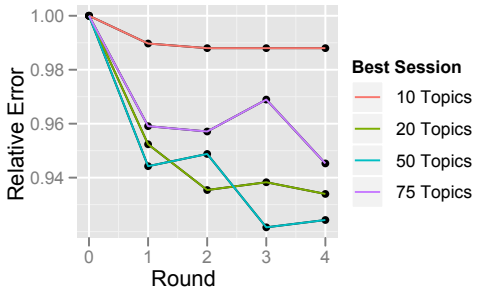

Figure 6: The relative error rate (using round 0 as a baseline) of the best Mechanical Turk user session for each of the four numbers of topics. While the 10-topic model does not provide enough flexibility to create good constraints, the best users could clearly improve classification with more topics.

matched specific newsgroups included religion, space exploration, graphics, and encryption. Other common themes were broader than individual newsgroups (e.g. sports, government and computers). Others matched sub-topics of a single newsgroup, such as homosexuality, Israel or computer programming.

Some users created inscrutable constraints, like ("better, people, right, take, things") and ("fbi, let, says"). They may have just clicked random words to finish the task quickly. While subsequent users could delete poor constraints, most chose not to. Because we wanted to understand broader behavior we made no effort to squelch such responses.

The two-word constraints illustrate an interesting contrast. Some pairs are linked together in the corpus, like ("jesus, christ") and ("solar, sun"). With others, like ("even, number") and ("book, list"), the users seem to be encouraging collocations to be in the same topic. However, the collocations may not be in any document in this corpus. Another user created a constraint consisting of male first names. A topic did emerge with these words, but the rest of the words in that topic seemed random, as male first names are not likely to co-occur in the same document.

Not all sensible constraints led to successful topic changes. Many users grouped "mac" and "windows" together, but they were almost never placed in the same topic. The corpus includes separate newsgroups for Macintosh and Windows hardware, and divergent contexts of "mac" and "windows" overpowered the prior distribution.

The constraint size ranged from one word to over 40. In general, the more words in the constraint, the more likely it was to noticeably affect the topic distribution. This observation makes sense given our ablation method. A constraint with more words will cause the topic assignments to be reset for more documents.

8 Discussion

In this work, we introduced a means for end-users to refine and improve the topics discovered by topic models. ITM offers a paradigm for non-specialist consumers of machine learning algorithms to refine models to better reflect their interests and needs. We demonstrated that even novice users are able to understand and build constraints using a simple interface and that their constraints can improve the model's ability to capture the latent structure of a corpus.

As presented here, the technique for incorporating constraints is closely tied to inference with Gibbs sampling. However, most inference techniques are essentially optimization problems. As long as it is possible to define a transition on the state space that moves from one less-constrained model to another more-constrained model, other inference procedures can also be used.

We hope to engage these algorithms with more sophisticated users than those on Mechanical Turk to measure how these models can help them better explore and understand large, uncurated data sets. As we learn their needs, we can add more avenues for interacting with topic models.

Acknowledgements

We would like to thank the anonymous reviewers, Edmund Talley, Jonathan Chang, and Philip Resnik for their helpful comments on drafts of this paper. This work was supported by NSF grant #0705832. Jordan Boyd-Graber is also supported by the Army Research Laboratory through ARL Cooperative Agreement W911NF-09-2-0072 and by NSF grant #1018625. Any opinions, findings, conclusions, or recommendations expressed are the authors' and do not necessarily reflect those of the sponsors.

References

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

David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jordan Boyd-Graber and David M. Blei. 2008. Syntactic topic models. In *Proceedings of Advances in Neural Information Processing Systems*.

Jordan Boyd-Graber and Philip Resnik. 2010. Holistic sentiment analysis across languages: Multilingual supervised latent Dirichlet allocation. In *Proceedings of Emperical Methods in Natural Language Processing*.

Jordan Boyd-Graber, David M. Blei, and Xiaojin Zhu. 2007. A topic model for word sense disambiguation. In *Proceedings of Emperical Methods in Natural Language Processing*.

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

Jonathan Chang. 2010. Not-so-latent Dirichlet allocation: Collapsed Gibbs sampling using human judgments. In *NAACL Workshop: Creating Speech and Language Data With Amazon'ss Mechanical Turk*.

Hal Daumé III. 2009. Markov random topic fields. In *Proceedings of Artificial Intelligence and Statistics*.

Laura Dietz, Steffen Bickel, and Tobias Scheffer. 2007. Unsupervised prediction of citation influences. In *Proceedings of International Conference of Machine Learning*, pages 233–240.

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

Amit Gruber, Michael Rosen-Zvi, and Yair Weiss. 2007. Hidden topic Markov models. In *Artificial Intelligence and Statistics*.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1):10–18.

Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence*.

Thomas K. Landauer, Danielle S. McNamara, Dennis S. Marynick, and Walter Kintsch, editors. 2006. *Probabilistic Topic Models*. Laurence Erlbaum.

Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on? identifying perspectives at the document and sentence levels. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*.

Edward Loper and Steven Bird. 2002. NLTK: the natural language toolkit. In *Tools and methodologies for teaching*. ACL.

Radford M. Neal. 1993. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto.

David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. 2010. Automatic evaluation of topic coherence. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

Michael Paul and Roxana Girju. 2010. A two-dimensional topic-aspect model for discovering multi-faceted topics. In *Association for the Advancement of Artificial Intelligence*.

James Petterson, Smola Alex, Tiberio Caetano, Wray Buntine, and Narayanamurthy Shravan. 2010. Word features for latent Dirichlet allocation. In *Neural Information Processing Systems*, December.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of Emperical Methods in Natural Language Processing*.

Fergus Rob, Li Fei-Fei, Perona Pietro, and Zisserman Andrew. 2005. Learning object categories from Google's image search. In *International Conference on Computer Vision*.

Michal Rosen-Zvi, Thomas L. Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *Proceedings of Uncertainty in Artificial Intelligence*.

Suyash Shringarpure and Eric P. Xing. 2008. mStruct: a new admixture model for inference of population structure in light of both genetic admixing and allele mutations. In *Proceedings of International Conference of Machine Learning*.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *Proceedings of Emperical Methods in Natural Language Processing*.

Hanna Wallach, David Mimno, and Andrew McCallum. 2009. Rethinking LDA: Why priors matter. In *Proceedings of Advances in Neural Information Processing Systems*.

Hanna M. Wallach. 2006. Topic modeling: Beyond bag-of-words. In *Proceedings of International Conference of Machine Learning*.

Limin Yao, David Mimno, and Andrew McCallum. 2009. Efficient methods for topic model inference on streaming document collections. In *Knowledge Discovery and Data Mining*.