Fast Inference for Interactive Models of Text

Jeffrey Lund, Paul Felt, Kevin Seppi, Eric K. Ringger
Department of Computer Science
Brigham Young University
{jefflund, paul_felt, kseppi}@byu.edu, ringger@cs.byu.edu

Abstract

Probabilistic models are a useful means for analyzing large text corpora. Integrating such models with human interaction enables many new use cases. However, adding human interaction to probabilistic models requires inference algorithms which are both fast and accurate. We explore the use of Iterated Conditional Modes as a fast alternative to Gibbs sampling or variational EM. We demonstrate superior performance both in run time and model quality on three different models of text including a DP Mixture of Multinomials for web search result clustering, the Interactive Topic Model, and MOMRESP, a multinomial crowdsourcing model.

1 Introduction

One of the most popular and useful approaches for analysis of large bodies of text documents is probabilistic models. For example, topic models such as Latent Dirichlet Allocation (LDA) can automatically learn topics from a set of documents, giving users a glimpse into the common themes of the data (Blei et al., 2003). Other models such as the Mixture of Multinomials can be used to perform document clustering allowing users to automatically organize text data (Meila and Heckerman, 2001; Walker and Ringger, 2008).

We are interested in use cases for probabilistic models of text which include human interaction. For example, the Interactive Topic Model (ITM) is a topic model that extends LDA to allow the user to inject model constraints in the form of word groupings while the topics are being learned (Hu et al., 2011). By including the user in the training process rather than simply learning the topics offline, the user can fine-tune the resulting topic model to better suit individual user needs and to accommodate a user's domain knowledge. However, if the training algorithm is too slow, the delay between receiving user feedback and presenting the updated model will harm the interaction due to increased cognitive load. Consequently, we require an inference algorithm which is both fast enough to facilitate interaction, and maintains (or improves upon) the accuracy of existing inference techniques.

For models like LDA, we typically perform training by calculating *maximum a posteriori* estimates of the latent topic variables and parameters given observed document data, with the idea that the setting of topic variables and parameters which maximizes the posterior distribution will best explain the observed data. Although various exact methods exist, such as belief propagation (Pearl, 1988) and the junction tree algorithm (Koller and Friedman, 2009), the complexity of exact posterior inference on such models is NP-HARD in general, so we resort to various approximations in order to optimize the posterior distribution (Sontag and Roy, 2009; Cooper, 1990). Some popular algorithms for approximate posterior inference include Gibbs sampling and mean field variational inference.

Each of these approximate inference algorithms has some drawbacks. For example, while variational inference is often very fast, it makes simplifying assumptions about the posterior distribution which can seriously degrade the quality of solutions for certain models, such as Mixture of Multinomials (Walker, 2012). However, for other models such as LDA we can achieve good estimates very quickly (Asuncion et al., 2009). Gibbs sampling provably generates samples from the posterior distribution and unlike

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 2997–3006, Osaka, Japan, December 11-17 2016.

variational inference, it is theoretically able to explore the entire support of the posterior manifold. Unfortunately, any reasonable restriction on the run time of the sampler means that we will only be able to explore a localized area of the support. Consequently, for most uses of probabilistic models of text, practitioners run a sampler for a period of time in the hope of finding an area of high probability, and then use the final sample as an approximation for the mode. While for some models, such as Mixture of Multinomials, this technique gives very good results (Walker and Ringger, 2008; Rigouste et al., 2007; Zhong and Ghosh, 2005), the lack of a convergence criteria can make the technique too slow for applications which require user interaction. For example, (2004) found that LDA requires hundreds of iterations of sampling before the log-likelihood of the model stabilizes in distribution. In practice, many of the probabilistic models of text we are interested in require similar numbers of sampling iterations.

As an alternative to techniques which introduce strong assumptions for posterior inference (e.g., variational inference) or lack clear and timely convergence criteria (e.g., Gibbs sampling), we will examine the use of Iterated Conditional Modes or ICM (Besag, 1986; Wellner et al., 2004), This algorithm is able to quickly achieve locally optimal *maximum a posteriori* estimates.

In section 2, we will briefly describe the ICM algorithm and compare it with other existing techniques. Then in section 3 we will empirically examine the performance of ICM in the context of three very different probabilistic models of text which can be used interactively. We first show that ICM performs well in the context of a non-parametric model by experimenting with a Dirichlet Process Mixture of Multinomials applied to the problem of web search result clustering. We then turn our attention to the Interactive Topic Model (Hu et al., 2011) to show that ICM improves performance over the previously published Gibbs sampler. Finally, we use ICM in the context of MOMRESP, a probabilistic model designed to infer true document class labels from noisy crowdsourced judgments (Felt et al., 2014).

2 Iterated Conditional Modes

Suppose we are given a probabilistic model of text with observed data x and unobserved variables θ. For the purpose of this discussion, θ may represent any number of unobserved parameters and latent variables. These parameters and variables can be either continuous or discrete. Like Gibbs sampling, Iterated Conditional Modes (ICM) relies on the fact that while computing a posterior distribution of the form $p(\theta|x)$ may be intractable, computing the complete conditional for a single variable θ_i while holding fixed both x and the rest of the parameters $\theta_{\neg i}$ is feasible in models with local conjugacy. By using the tractable complete conditional distribution $p(\theta_i|\theta_{\neg i}, x)$ we are able to locally maximize the posterior without the need to approximate the posterior with samples.

The general procedure for ICM is very similar to Gibbs sampling. We cycle through each unobserved variable θ_i in the model and update current value of the variable to be the mode of its complete condition distribution. The ICM update is repeated until convergence when the value of each θ_i is already the mode of its complete conditional distribution.

To see that Iterated Conditional Modes will find a local maxima of the posterior distribution, we will demonstrate that the ICM updates monotonically increase the current estimate of the posterior probability $p(\theta|x)$. Since the data x is fixed, the posterior is proportional to the joint distribution over all of the variables and data:

$$p(\theta|x) = \frac{p(\theta, x)}{p(x)} \propto p(\theta, x) \tag{1}$$

Using the chain rule, for some i we can then factor the joint distribution as:

$$p(\theta, x) = p(\theta_i|\theta_{\neg i}, x) \cdot p(\theta_{\neg i}|x) \cdot p(x) \tag{2}$$

Since x is fixed, we can also write

$$p(\theta, x) \propto p(\theta_i|\theta_{\neg i}, x) \cdot p(\theta_{\neg i}|x) \tag{3}$$

While updating the parameter θ_i, $\theta_{\neg i}$ is held fixed, which means that the second term $p(\theta_{\neg i}|x)$ is constant and the factored joint distribution is proportional to the complete conditional, thus:

$$p(\theta|x) \propto p(\theta_i|\theta_{\neg i}, x) \tag{4}$$

Since these two expressions are proportional, setting θ_i to the mode of its complete conditional will only increase the value of the posterior probability. Thus our update rule for the variable θ_i is given as:

$$\hat{\theta}_i = \underset{k}{\operatorname{argmax}}\ p(\theta_i = k|\theta_{\neg i}, x) \tag{5}$$

Since this update equation monotonically increases the estimate of the posterior probability, and the value of the posterior probability is bounded above by 1, we can use the monotone convergence theorem to conclude that this algorithm will converge to a local maximum in the posterior distribution. Furthermore, the ICM algorithm is able to do so without approximating the whole posterior distribution.

Iterated Conditional Modes is related to Expectation Conditional Maximization (Meng and Rubin, 1993) in that it employs conditionals to find local maxima in a distribution. However, unlike Expectation Conditional Maximization which maximizes a likelihood, ICM is not a variant of EM, as it takes into account prior distributions when computing the complete conditionals. In fact, we could describe ICM as a particular limit of Gibbs sampling in the same way that K-means can be viewed as the deterministic limit of the EM algorithm (Bishop, 2006).

Note however that the efficiency of the ICM algorithm depends entirely on the ability to quickly compute the mode of the conditional distributions. If the complete conditional distribution is a density, this may involve continuous optimization. However, there is a wide class of models for computing the mode of the conditional distribution is easy. Typically this is done through the use of conjugate priors which make the conditional distributions tractable. For example, for many probabilistic models of text, computing the mode of the conditional distribution is often easy due to the frequent use of the Dirichlet-Multinomial conjugate pair.

To be clear, Iterated Conditional Modes is a coordinate ascent algorithm, and as such, it can only locally optimize the posterior — there is no guarantee of finding a global maximum. Thus if the posterior manifold contains many sub-par local maxima, then ICM will be sensitive to initialization and may perform poorly. Random restarts may mitigate the problem. Alternatively, an initialization strategy which consistently starts the inference procedure near a good solution will yield better *maximum a posteriori* estimates. The best initialization strategy depends on the model to be optimized. Thus each experiment described below includes its own initialization strategy.

3 Experiments

We now demonstrate that Iterated Conditional Modes converges quickly enough to allow for interactive use cases involving various probabilistic models of text, while yielding high quality estimates. In the hopes of demonstrating the general applicability of the technique, we do so on three different models and tasks. The first model is a DP Mixture of Multinomials applied to the task of web search result clustering. We choose this model to show that ICM can work in the context of non-parametric models. The second task is the Interactive Topic Model or ITM. We choose this model to suggest that ICM may be viable for a wide variety of interactive topic modeling applications. Finally, we will apply ICM to the MOMRESP model, which is a probabilistic model for producing annotated corpora for NLP and machine learning research.

3.1 Web Search Result Clustering

As many as 16% of queries issued to search engines contain ambiguous search terms (Song et al., 2007). After issuing a search query with this kind of ambiguity, users may become confused by seemingly unrelated search results, or they may be slowed by the need to narrow the scope of the query. For example, suppose a user issues the query "tiger." The user may be surprised to see results about the large feline, the golfer Tiger Woods and the German tanks used in the 1940s, when only one of those meanings of "tiger" was intended. Web search result clustering helps users deal with query ambiguity by automatically discovering clusters among the search results and presenting the results as clusters (Carpineto et al., 2009b). With a web search result clustering system, the user can select the cluster in which they are actually interested, and immediately filter out irrelevant results. An example of such a system is the Carrot[2]

search framework, which is available online both as a web service and as a downloadable application.*

Client-side web search result clustering systems do not maintain their own search index or data but instead rely on search results (specifically the snippets) returned from an external search engine chosen by the user. This allows users to utilize web search result clustering systems on a wide variety of search engines, both public and private. Since web search result clustering systems are meant to work with arbitrary search results, the computation typically takes place client-side. An important consequence is that web search result clustering systems should be able cluster extremely small amounts of data: rather than tens of thousands of full documents encountered in typical document clustering tasks, a web search result clustering systems uses around 100 documents, each of which consist of no more than a sentence or two. Furthermore, web search result clustering systems must be run quickly enough to facilitate web search. For simple interactions like issuing a web search query, the interaction take less than one second (Cook and Thomas, 2005). This stands in contrast to typical document clustering settings which can be run offline possibly using parallel computation resources rather than online using a single commodity machine. Due to these run time constraints, it has been argued that traditional document clustering techniques may not work out of the box (Carpineto et al., 2009b). Consequently, various specialized algorithms for the problem of web search result clustering have been published.

The best reported solution to the problem of web search result clustering employs maximal spanning trees to perform word sense induction (Di Marco and Navigli, 2011). The algorithm, referred to as MST, uses the Google Web1T n-gram data set (Brants and Franz, 2006) to create a co-occurrence graph on the words in the snippet results and then calculates maximal spanning trees to remove edges from the graph. This process repeats until the desired number of word clusters is formed. Unfortunately, the requirement of large amounts of n-gram data is not amenable to client-side computation. Even just maintaining an up-to-date n-gram data set (so that the system can handle queries related to fast-changing subjects such as recent popular culture) is also necessary but requires web-scale data-gathering resources. Consequently, MST is not a client-side solution.

There are, however, a number of approaches which are amenable to client-side web search result clustering. One such system is Lingo, which was developed for use in the Carrot^2 search framework (Osiński et al., 2004). Another is KEYSRC, which extracts key phrases from snippet data and then uses hierarchical agglomerative clustering on those phrases (Bernardini et al., 2009).

Despite the fact that model-based approaches tend to yield higher quality results in the general problem of document clustering (Zhong and Ghosh, 2005), no study has applied model-based clustering to the specialized problem of web search result clustering. We rectify the lacuna by applying Iterated Conditional Modes to a Dirichlet Process Mixture of Multinomials model (hereafter referred to as DP-MOM), and we compare our results to those of the previously studied web search result clustering solutions.

Our model-based approach employs a Dirichlet Process (DP) mixture model, a well studied Bayesian non-parametric model (Antoniak and others, 1974; Neal, 2000). This type of model has been used to perform document clustering, albeit with modifications to include feature selection in the model (Yu et al., 2010). Given the scarcity of data in this application, we cannot realistically perform feature selection (although the snippet generation itself might be viewed as feature selection).

DP mixture models have a known relationship with the Chinese Restaurant Process (CRP) in that if we integrate over the random mixing measure in the DP mixture, the resulting model will have a CRP prior over the mixture components (Blackwell and MacQueen, 1973). Taking advantage of this relationship, the DP-MOM model can be written with the following form:

$$\phi_k|\beta \sim Dirichlet(\beta), k = 1, ..., K$$

$$z_d|\alpha \sim CRP(\alpha), d = 1, ..., M$$

$$w_d|z_d, \phi \sim Multinomial(N_d, \phi_{z_d}), d = 1, ..., M$$

where ϕ_k is the word distribution for topic k with a symmetric Dirichlet prior of β, z_d is the cluster assignment of document d, α is CRP concentration, and w_d gives the observed token counts for document d. M is the number of documents, and N_d is the number of tokens in the dth document.

*http://search.carrot2.org

Algorithm	AMBIENT	MORESQUE	All
ICM DP-MOM	**.768**	**.570**	**.625**
Gibbs DP-MOM	.758	.544	.604
KEYSRC	.665	.558	.588
Lingo	.628	.527	.555
MST	.815	.867	.852

Table 1: Clustering quality results, as measured by the Rand index. We include the MST results for reference though they do not constitute client-side results. Bold indicates the best client-side result, and underline indicates the absolute best result.

Following the advice of (2000), we derive a collapsed Gibbs sampler by integrating over ϕ. The complete conditional probability for the cluster assignment z_d, given the other assignments $z_{\neg d}$ and data is

$$p(z_d = j | z_{\neg d}, w) \propto \begin{cases} c_j \prod_{v \in V} \frac{\Gamma(\beta + n_{jv} + w_{dv})}{\Gamma(|V|\beta + n_{j\cdot} + w_{d\cdot})} \frac{\Gamma(|V|\beta + n_{j\cdot})}{\Gamma(\beta + n_{jv})}, & \text{if } c_j > 0 \\ \alpha \prod_{v \in V} \frac{\Gamma(\beta + w_{dv})}{\Gamma(|V|\beta + w_{d\cdot})} \frac{\Gamma(|V|\beta)}{\Gamma(\beta)}, & \text{otherwise} \end{cases} \tag{6}$$

where V is the set of words in the data, c_j is the count of documents assigned to cluster j, n_{jv} is the number of times the word type v is present in a document assigned to cluster j, and w_{dv} is the number of times word v is found in document d. Dots in the subscripts of these counters indicate marginalization over the missing index. For the sake of space, we omit the derivation, but it is similar to the derivation for the finite Mixture of Multinomials given by (2008).

We can also derive a mean field variational inference algorithm for DP-MOM. Such an algorithm, while fast, yields extremely poor *maximum a posteriori* estimates for DP-MOM (Walker, 2012). At least for this model, the independences introduced by the mean field assumption are too strong. Consequently, we compare ICM to a baseline Gibbs sampler instead of variational inference.

The final detail needed for implementing DP-MOM is an initialization strategy. A key advantage of using a non-parametric model is that the model can learn the number of clusters from data, thereby allowing our model to perform well with varying amounts of query term ambiguity. We can either initialize with a large number of clusters and let the model shrink to fit the data or to start with a small number of clusters and grow to fit the data. Our experiments indicate that starting with a single cluster performed the best, so we utilized this initialization strategy for our results.

In order to validate that our model-based approach performs well, we follow the same methodology as Di Marco and Navigli (2011) when evaluating the MST algorithm. We experiment with two different datasets: AMBIENT (Carpineto et al., 2009a) and MORESQUE (Di Marco and Navigli, 2011). Each dataset is a set of search queries issued to the YAHOO! search engine, along with the top 100 search result snippets which have all been manually labeled with topics. The primary difference between the two datasets is that the earlier AMBIENT dataset consists of single word queries, while the MORESQUE dataset extends AMBIENT to queries of length 2–4[†]. Taking both datasets together, we have a total of 158 ambiguous queries, each with between 3 and 15 topics.

Still following Di Marco and Navigli (2011), we evaluate clustering performance on these ambiguous query datasets with two metrics. The first is the Rand Index (Rand, 1971), a measure of similarity between two clusterings over the same set of elements. The Rand Index can be viewed a kind of accuracy, since it gives the percentage of pairing decisions which were correctly made with respect to a base clustering.

Table 1 summarizes the results of the various web search clustering algorithms measured by Rand Index. We see that the MST algorithm performs the best, but we remind the reader that this algorithm far exceeds the computation resource requirements for client-side web search result clustering so it only serves as a baseline. Among previously studied algorithms which respect resource constraints, our model-based approach outperforms existing techniques by a wide margin. Interestingly, Iterated Conditional Modes outperforms Gibbs sampling, indicating that for this model and this task, the extra exploration within

[†]Note that the data we use to cluster is the resulting snippets, *not* the queries themselves.

Algorithm	K=3	K=5	K=10	K=15	K=20
ICM DP-MOM	**.517**	**.650**	**.811**	**.885**	**.927**
Gibbs DP-MOM	.508	.635	.795	.873	.917
YAHOO!	.492	.600	.729	.785	.827
KEYSRC	.443	.558	.720	.791	.832
MST	.547	.656	.792	.867	.907

Table 2: Diversification results on all queries, as measured by S-recall@K. The ordering provided by the YAHOO! search engine is included as a baseline. Bold indicate the best client-side result, and underline indicates the best absolute result.

a region of high probability from sampling is not as important as the ability to jump to a mode in that region of high probability. Furthermore, our approach is extremely fast. Using a single core of an AMD Phenom II X6 1090T processor, the median time spent using ICM to perform clustering on results for a single query was 1.18 milliseconds. Our experiments using Gibbs sampling on the other hand took 6.78 milliseconds to complete.

Our second measure evaluates the diversification produced by a clustering algorithm. As outlined by Di Macro and Navigli (2011), we can use the clustering labels to re-rank the search results such that the top search results are more diverse. We measure the diversification with S-recall@K, which measures the percentage of ground-truth labeled topics present in the top K search results after re-ranking. We use the ordering returned by YAHOO! as a baseline for S-recall@K.

Table 2 shows the results of the various web search result clustering algorithms with respect to S-recall@K. In both cases, our model-based approach outperforms the baseline ranking. Both KEYSRC and Lingo actually did worse than the YAHOO! baseline. For $K \leq 5$, MST performs the best. However, for all other values of K, our model-based approach performs the best. This is likely due to the fact that our non-parametric model is able to increase the number of clusters in the presence of highly ambiguous queries, whereas the MST algorithm uses a pre-specified number of clusters.

3.2 Interactive Topic Model

We now turn our attention to the Interactive Topic Model or ITM (Hu et al., 2011). This model extends LDA by replacing the per-topic categorical distributions over words with a tree-structured Dirichlet-forest distribution. The user interactively injects constraints into the model by placing token types into Dirichlet trees. Depending on the prior for the Dirichlet trees, the constraint can either be a "must link" (positive correlation) or a "cannot link" (negative correlation) type of constraint (Andrzejewski et al., 2009). Due to issues with transitivity in "cannot link" constraints, we follow Hu et al. (2011) and focus on "must link" constraints by setting the Dirichlet trees parameter to be extremely high. A user is able to employ constraints to tell the model to give a particular set of word types similar probability within each individual topic. For the sake of brevity, we omit details about the ITM including the specific distributions for the model and the complete conditionals used to drive the collapsed Gibbs sampler for the model since they are thoroughly explained by Hu et al. (2011). We also note that variational inference is inappropriate for this model, as it only achieves good performance when used in conjunction with hyper-parameter optimization. However, such optimization tends to undo the constraints, rendering the model useless (Hu et al., 2011). Consequently, we use the published Gibbs sampler as our baseline inference algorithm.

The ITM model is trained as follows: first we train a base model with no constraints (equivalent to learning a vanilla LDA model). The user is then presented with the outcome of an analysis using the model, possibly by showing them the traditional topic lists wherein a topic is represented by the most probable words in the topic. The user then injects word constraints into the model according to the individual needs of the user or specific domain knowledge. Using the document-level ablation strategy recommended by Hu et al. (2011) the topic assignments of any document which contains a newly constrained word are revoked. In order to enforce model consistency, the rest of the topic assignments remain unchanged. Finally, inference is rerun with the new constraints and the updated model is presented to the user. This interactive process is repeated until the user is satisfied with the final state of the model.

We now investigate which inference algorithm performs the interactive model updates best. Cook and

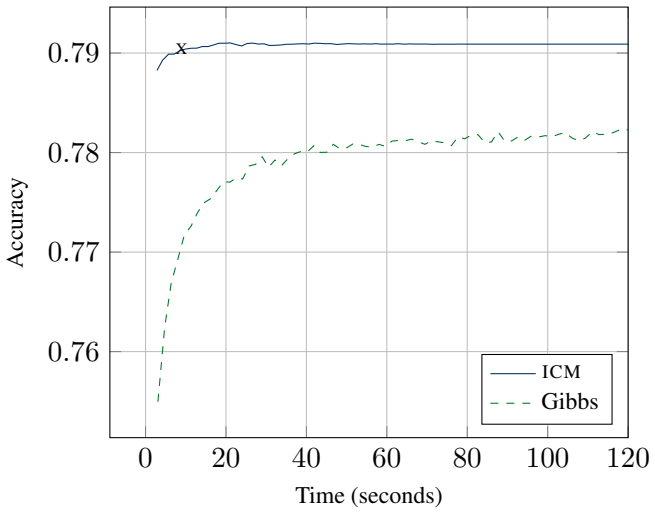

Figure 1: Accuracy versus time for both Iterated Conditional Modes and Gibbs sampling with the ITM. The X-mark indicates the median point of convergence for ICM.

Thomas (2005) show that for a complex user-initiated activity such as requesting a model update, we require a response time which is ten seconds or less or the human-computer interaction may suffer. Unfortunately, if we perform the recommended 30 iterations of sampling, even at one second per iteration, this can be taxing on the user. We turn to ICM as a faster alternative.

In order to validate the performance of Iterated Conditional Modes on the ITM, we employ an experimental setup similar to that of (2011) using the well-known 20 Newsgroups corpus, which consists of roughly 20,000 documents divided into 20 newsgroups. We simulate a user's constraints by selecting words using information gain with respect to the newsgroup labels. After training a base model with 100 iterations of Gibbs sampling for burn-in, we inject the simulated constraints into the model. Finally, we run inference using either ICM or Gibbs sampling from this point. We evaluate the model quality with a classification task in which we train a classifier to predict the source newsgroup of an unlabeled document using topic-word features. To do so, the corpus is split into a training and test set, and each word along with its assigned topic is used as a feature for the classifier. We report the classification accuracy from a support vector machine trained on the topic-word pairs from the documents in the training set. As with Hu et al. (2011), we do not hope to achieve state-of-the-art classification results for this dataset, but we do hope that the classification trends will demonstrate which inference algorithm better drives the model towards the original (withheld) human labels once the simulated constraints have been added.

As shown in Figure 1 Iterated Conditional Modes outperforms Gibbs sampling. This indicates that there is more value in reaching a local maximum than there is in the exploration that comes from sampling. More importantly, ICM has the potential to run much faster than Gibbs sampling: rather than running a Gibbs sampler for the recommended 30 iterations, the median number of iterations required for ICM to converge was 9, which allows us to present the updated model to the user within the ten second time frame recommended by Cook and Thomas (2005).

3.3 MOMRESP

Microtask markets such as Amazon's Mechanical Turk (mturk.com) allow corpora to be labeled at extremely low cost, a practice known as crowdsourcing. However, the recent emergence of crowdsourcing as the preferred method for labeling document corpora has introduced an important research problem: how to mitigate the inaccuracy of crowdsourced judgments. A common solution is to obtain multiple redundant judgments, or annotations, and aggregate them using a baseline strategy such as *majority vote*.

When annotations are both plentiful and highly accurate, majority vote works well. However, crowdsourced annotations are seldom highly accurate. State-of-the-art solutions are model-based and use standard inference algorithms. For example, the MOMRESP model presented by Felt et al. (2014) describes

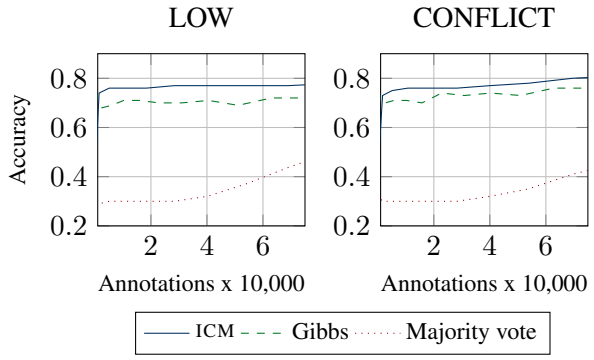

Figure 2: Accuracy of inferred labels versus the number of annotations given to the model. At the last plotted point each document has on average nearly 4 annotations. Gibbs and ICM use MOMRESP. A majority vote baseline is also shown for reference.

a joint model over document features and annotations. We include here a sketch of the model, deferring details to the referenced paper. Documents and annotations are both modeled as count vectors with multinomial distributions conditioned on the true but unobserved class label. Parameters include both per-class word distributions and class confusion matrices for each annotator. When annotations are scarce or of low-quality, the MOMRESP model trained with Gibbs sampling significantly outperforms majority vote in terms of inferred label accuracy. Labels inferred by Iterated Conditional Modes are even more accurate.

In order to validate this claim, we run MOMRESP with both Gibbs and ICM on synthetic annotations produced for the 20 newsgroups dataset. We draw synthetic annotators from the LOW and CONFLICT annotator pools described by (2014). Each pool consists of 5 annotators. In both pools, annotators give correct judgments with probabilities .5, .4, .3, .2, .1, respectively. In the LOW pool, annotator errors are distributed uniformly across incorrect classes. In the simulated CONFLICT pool, errors are systematic: a confusion matrix is created for each annotator whose diagonal is set to the annotator's accuracy and whose off-diagonal row entries are sampled from a symmetric Dirichlet distribution with parameter 0.1, to encourage sparsity, and then scaled so that each row sums to 1. CONFLICT errors are produced by corrupting true labels according to this confusion matrix. Documents are annotated in random order without replacement, and after all documents have one annotation, the process is repeated. Simulated annotation continues until we have reached the desired number of annotations. We then initialize MOMRESP using majority vote to set initial class label values and perform posterior inference using both Gibbs and ICM. We compare the model-inferred class labels with the gold standard class labels for each document in order to compute model accuracy.

Figure 2 plots the inferred label accuracy of Gibbs and ICM as well as majority vote for reference. Regardless of the number of annotations, ICM yields better accuracy than Gibbs sampling for both the LOW and the CONFLICT cases. While not shown, this trend hold even cases where majority vote outperforms MOMRESP.

In addition to inferring more accurate document labels than Gibbs, ICM has a run time which is orders of magnitude faster than that of Gibbs sampling. The median time of convergence was 6.72 seconds. This falls well within the run time recommended by Cook and Thomas (2005) for complex user-initiated tasks such as rerunning inference on MOMRESP given additional annotations. Consequently, if MOMRESP were to be adapted for an active learning task, then ICM would provide not only accurate posterior inference, but run times which are amenable to active learning.

4 Conclusion

Iterated Conditional Modes is a coordinate ascent algorithm that yields locally optimal *maximum a posteriori* estimates for models with tractable complete conditionals. We have shown that ICM identifies *maximum a posteriori* solutions that are superior to those found by Gibbs sampling for three applica-

tions: web search result clustering, topic modeling, and crowdsourcing problems. In addition, Iterated Conditional Modes has termination criterion which is easily identified, while it can be difficult to determine when a Gibbs sampler has reached the stationary distribution. Because of the convergence of ICM, we were able to significantly speed up the three applications compared to Gibbs sampling, enabling better human interactivity. These experiments motivate further exploration of this inference technique, particularly in interactive use cases of models in which both run time and model quality are crucial.

Acknowledgements

This work was supported by the NSF Grant IIS-1409739. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

References

David Andrzejewski, Xiaojin Zhu, and Mark Craven. 2009. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–32. ACM.

Charles E Antoniak et al. 1974. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174.

Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee Whye Teh. 2009. On smoothing and inference for topic models. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 27–34. AUAI Press.

Andrea Bernardini, Claudio Carpineto, and Massimiliano D'Amico. 2009. Full-subtopic retrieval with keyphrase-based search results clustering. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 1, pages 206–213. IET.

Julian Besag. 1986. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302.

Christopher M Bishop, 2006. *Pattern recognition and machine learning*, volume 1, pages 443–444. Springer New York.

David Blackwell and James B MacQueen. 1973. Ferguson distributions via pólya urn schemes. *The Annals of Statistics*, pages 353–355.

David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent dirichlet allocation. *JMLR*, 3:993–1022.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram version 1.

Claudio Carpineto, Stefano Mizzaro, Giovanni Romano, and Matteo Snidero. 2009a. Mobile information retrieval with search results clustering: Prototypes and evaluations. *Journal of the American Society for Information Science and Technology*, 60(5):877–895.

Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. 2009b. A survey of web clustering engines. *ACM Computing Surveys (CSUR)*, 41(3):17.

Kristin A. Cook and James J. Thomas. 2005. Illuminating the path: The research and development agenda for visual analytics. Technical report, Pacific Northwest National Laboratory (PNNL), Richland, WA (US).

Gregory F Cooper. 1990. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence*, 42(2):393–405.

Antonio Di Marco and Roberto Navigli. 2011. Clustering web search results with maximum spanning trees. In *AI* IA 2011: Artificial Intelligence Around Man and Beyond*, pages 201–212. Springer.

Paul Felt, Robbie Haertel, Eric Ringger, and Kevin Seppi. 2014. Momresp: A bayesian model for multi-annotator document labeling. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May. European Language Resources Association (ELRA).

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

Yuening Hu, Jordan L Boyd-Graber, and Brianna Satinoff. 2011. Interactive topic modeling. In *ACL*, pages 248–257.

Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

Marina Meila and David Heckerman. 2001. An experimental comparison of model-based clustering methods. *Machine learning*, 42(1-2):1–2.

Xiao-Li Meng and Donald B Rubin. 1993. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278.

Radford M Neal. 2000. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265.

Stanisław Osiński, Jerzy Stefanowski, and Dawid Weiss. 2004. Lingo: Search results clustering algorithm based on singular value decomposition. In *Intelligent information processing and web mining*, pages 359–368. Springer.

Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Loïs Rigouste, Olivier Cappé, and François Yvon. 2007. Inference and evaluation of the multinomial mixture model for text clustering. *Information processing & management*, 43(5):1260–1280.

Ruihua Song, Zhenxiao Luo, Ji-Rong Wen, Yong Yu, and Hsiao-Wuen Hon. 2007. Identifying ambiguous queries in web search. In *Proceedings of the 16th international conference on World Wide Web*, pages 1169–1170. ACM.

David Sontag and Daniel M Roy. 2009. Complexity of inference in topic models. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*.

Daniel David Walker and Eric K. Ringger. 2008. Model-based document clustering with a collapsed gibbs sampler. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 704–712. ACM.

Daniel David Walker. 2012. *Bayesian text analytics for document collections*. Ph.D. thesis, Brigham Young University.

Ben Wellner, Andrew McCallum, Fuchun Peng, and Michael Hay. 2004. An integrated, conditional model of information extraction and coreference with application to citation matching. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 593–601. AUAI Press.

Guan Yu, Ruizhang Huang, and Zhaojun Wang. 2010. Document clustering via dirichlet process mixture model with feature selection. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 763–772. ACM.

S. Zhong and J. Ghosh. 2005. Generative model-based document clustering: a comparative study. *Knowledge and Information Systems*, 8(3):374–384.