Topic Models for Taxonomies

Anton Bakalov, Andrew McCallum,
Hanna Wallach
Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003
{abakalov,mccallum,wallach}@cs.umass.edu

David Mimno
Dept. of Computer Science
Princeton University
Princeton, NJ 08540
mimno@cs.princeton.edu

ABSTRACT

Concept taxonomies such as MeSH, the ACM Computing Classification System, and the NY Times Subject Headings are frequently used to help organize data. They typically consist of a set of concept names organized in a hierarchy. However, these names and structure are often not sufficient to fully capture the intended meaning of a taxonomy node, and particularly non-experts may have difficulty navigating and placing data into the taxonomy. This paper introduces two semi-supervised topic models that automatically augment a given taxonomy with many additional keywords by leveraging a corpus of multi-labeled documents. Our experiments show that users find the topics beneficial for taxonomy interpretation, substantially increasing their cataloging accuracy. Furthermore, the models provide a better information rate compared to Labeled LDA [7].

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; H.3.7 [**Information Systems**]: Digital Libraries; G.3 [**Mathematics of Computing**]: Probability and Statistics—*Statistical computing*

Keywords

Topic modeling, Taxonomy annotation, Taxonomy browsing

1. INTRODUCTION

Many organizations such as the Association of Computing Machinery (ACM) use taxonomies of classes as structured metadata to facilitate browsing of their document libraries. Users browse and search these libraries by navigating a hierarchy of named concept nodes. When new documents are added to the library they must also be assigned one or more concept names. This task is often performed by the document authors themselves, not trained catalogers.

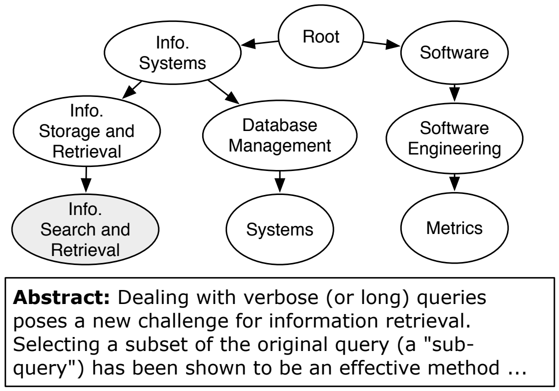
Unfortunately, the concept node names often do not describe the concept in sufficient detail for unfamiliar users to fully understand the topics a node is intended to capture. We present a user study (Section 4.3) confirming that substantial inaccuracies arise when asking computer science graduate students to assign a research paper to nodes of the ACM taxonomy when given its node names and hierarchical structure. Users could gain better understanding by reading titles of papers that have previously been accurately assigned, but this would be time consuming. Taxonomy builders could augment each concept name with a list of keywords that delineate the concept. But this task would be burdensome to perform manually for large taxonomies, and furthermore would need to be redone frequently since new ideas and topics within a concept arise over time.

This paper presents two semi-supervised topic models that automatically discover lists of relevant keywords for taxonomic concepts. The models, termed Labeled Pachinko Allocation (LPAM) and LPAM-List, take as input an existing taxonomy as well as documents with their concept assignments. Then they run inference in a latent-Dirichlet-allocation-like manner in which there is one topic per concept node, and the set of topics allowable in the document is restricted by its taxonomic concept labels, augmented with their ancestors in the hierarchy. Document terms are assigned to the topics, and the highest weighted words in each topic become the concept keywords. Notably, even though most documents are labeled with multiple concepts and all concepts pull in their ancestors, the model is able to partition the keywords into their appropriate concepts. Furthermore, incorporating ancestor nodes enables our models (a) to discover keywords for more abstract concepts located close to the root of the taxonomy, and (b) to separate more generic words out of the taxonomy leaves.

Multiple previous papers have focused on learning topic hierarchies (*e.g.*, [2, 4]). We are addressing the complementary problem—leveraging a given human-defined hierarchy such as ACM's Computing Classification System. Like our work, some other methods sample paths to nodes in a given taxonomy (*e.g.*, Hierarchical Concept Topic Model (HCTM) [9], Hierarchical Pachinko Allocation (HPAM) [5], Multilingual Supervised LDA (ML-SLDA) [3]). However, LPAM leverages available labels to select a subtree of the taxonomy to generate a given document. Furthermore, LPAM differs from HCTM in that each node has a distribution over the whole vocabulary, not over a pre-specified subset.

The second model we propose, LPAM-List, represents a document as a mixture of the same topic nodes as LPAM's, but does not use the tree structure in its generative process. This makes it more similar to Labeled LDA [7] which also

JCDL'12, June 10–14, 2012, Washington, DC, USA.



Abstract: Dealing with verbose (or long) queries poses a new challenge for information retrieval. Selecting a subset of the original query (a "sub-query") has been shown to be an effective method ...

Figure 1: The first two sentences from the abstract of a paper about information retrieval [11]. The publication is labeled with the shaded node from the ACM taxonomy. Some of the neighboring nodes are also displayed.

constrains the choice of topics to those associated with the document's assigned concepts. However, LPAM-List additionally incorporates the taxonomic ancestors of the assigned nodes. This simple change enables it to learn about both leaves and interior nodes of the taxonomy. Similarly to Labeled LDA, Newman et. al. [6] also uses a non-hierarchical set of concept labels, but rather than discovering a distribution of words for each concept, they instead estimate a distribution of topics for each concept; this is accomplished by using document labels as authors in the author-topic model [8].

Our experiments show that leveraging the internal structure of taxonomies imparts our methods with two advantages. First, we obtain a better information rate compared to Labeled LDA (Section 4.2), indicating that our models more precisely capture characteristics of the data. Second, we obtain a list of keywords for each concept in the taxonomy, including the high-level interior concepts that do not usually appear as labels. Our user study shows that our concept description keywords help people interpret a taxonomy as measured by the accuracy of concept label assignments (Section 4.3).

2. MODELS

2.1 Labeled Pachinko Allocation

LPAM is a semi-supervised topic model for documents labeled with nodes from a taxonomy. It builds on latent Dirichlet allocation (LDA) [1] and hierarchical pachinko allocation [5] by leveraging additional structure, as described in this section. For each document d, it considers a restricted set of taxonomy nodes C_d consisting of the document labels, augmented with their ancestors in the hierarchy. For example, the document in Figure 1 is a mixture of the nodes Info. Search and Retrieval, Info. Storage and Retrieval, Info. Systems, and Root. Each node $c \in C_d$ defines a multinomial distribution $\boldsymbol{\theta}^{(d,c)}$ over child nodes in C_d plus an additional one termed the *exit*. Also, each node r in the taxonomy is associated with a multinomial distribution $\boldsymbol{\phi}^{(r)}$ over the vocabulary. Like some other models such as HPAM, LPAM represents a distribution over paths using multinomials over child nodes; however, LPAM additionally enforces path constraints based on labels.

LPAM's generative process operates as follows:
1. For each node r draw $\boldsymbol{\phi}^{(r)} \sim \mathrm{Dir}(\boldsymbol{\beta})$
2. For each document d:
(a) For each node $c \in C_d$, draw $\boldsymbol{\theta}^{(d,c)} \sim \mathrm{Dir}(\boldsymbol{\alpha}^{(d,c)})$.
(b) For each word w:
- Draw $r \sim \mathrm{Mult}(\boldsymbol{\theta}^{(d,root)})$
- While r is not an *exit*, draw $r \sim \mathrm{Mult}(\boldsymbol{\theta}^{(d,r)})$
- Draw $w \sim \mathrm{Mult}(\boldsymbol{\phi}^{(a)})$ where a is r's parent

We train the models by Gibbs sampling. The probability of choosing node c as the topic of the i'th token (w_i) given the remaining topic assignments ($\mathbf{z}_{\setminus i}$) and all words ($\mathbf{w}$) is:

$$P(z_i = c|\mathbf{z}_{\setminus i}, \mathbf{w}, \mathbf{U}) \propto \left(\prod_{j=2}^{k} \frac{n^{(d)}_{c_{j-1},c_j} + \alpha^{(d,c_{j-1})}_{c_j}}{\sum_r (n^{(d)}_{c_{j-1},r} + \alpha^{(d,c_{j-1})}_r)}\right) \frac{n^{(w)}_c + \beta_w}{\sum_m (n^{(m)}_c + \beta_m)}$$

where $\mathbf{U}$ is the set of hyperparameters; $n^{(d)}_{c_{j-1},c_j}$ is the number of times node c_j is visited from its parent c_{j-1} for document d; $n^{(w)}_c$ is the number of times word w is assigned to node c; k is the level at which the *exit* is located, *i.e.*, c_{k-1} is the node generating the word. The contribution of the token being sampled is removed from these counts.

The first term of the right-hand side expression above is the probability of traversing the path to node c and choosing to emit from it. The second term is the probability of generating word w from the word distribution associated with concept c.

2.2 Labeled Pachinko Allocation - List

LPAM-List considers the same restricted set of topics C_d as LPAM's but sampling a path is not part of the generative process. If $\boldsymbol{\psi}^{(d)}$ is a multinomial distribution over the nodes in C_d, then the generative storyline is:
1. For each node r draw $\boldsymbol{\phi}^{(r)} \sim \mathrm{Dir}(\boldsymbol{\beta})$
2. For each document d:
(a) Draw $\boldsymbol{\psi}^{(d)} \sim \mathrm{Dir}(\boldsymbol{\alpha}^{(d)})$
(b) For each word w draw $c \sim \mathrm{Mult}(\boldsymbol{\psi}^{(d)})$ and $w \sim \mathrm{Mult}(\boldsymbol{\phi}^{(c)})$

In contrast, Labeled LDA [7] constrains the topics only to those associated with the document labels. LPAM-List's sampling equation is:

$$P(z_i = c|\mathbf{z}_{\setminus i}, \mathbf{w}, \boldsymbol{\alpha}^{(d)}, \boldsymbol{\beta}) \propto (n^{(d)}_c + \alpha^{(d)}_c) \frac{n^{(w)}_c + \beta_w}{\sum_m (n^{(m)}_c + \beta_m)}$$

where z_i is the topic of the i'th token w_i; $\mathbf{z}_{\setminus i}$ are the topic assignments of the remaining tokens; $c \in C_d$; $n^{(d)}_c$ is the number of times node c is sampled in document d; $n^{(w)}_c$ is the number of times word w was assigned to node c. Similarly to LPAM's sampling equation, the contribution of the token being sampled is removed from these counts.

3. DATASETS AND PREPROCESSING

We perform experiments on two datasets: (1) abstracts of scientific publications from the ACM digital library, and (2) the first 300 tokens of articles from the New York Times (NYT) dataset. We remove stopwords and tokens that appear fewer than five times in the corresponding corpus.

Statistics	**ACM**	**NYT**
Taxonomy size	268	580
Taxonomy depth	4	6
Labels/doc	3.13	5.98
Nodes/doc	8.2	8.94
Branching nodes/doc	1.65	2.11
Number of docs	20,527	15,493
Vocabulary size	13,906	27,544
Num tokens	1,499,878	3,323,851
Tokens/doc	73.07	214.54

Table 1: Dataset statistics. *Taxonomy size* is the number of nodes in the taxonomy. (We exclude nodes that are not used to label any document in our dataset and are not ancestors of label nodes. These concepts do not affect sampling and it is impossible to discover keywords for them.) *Labels/doc* is the average number of document tags. *Nodes/doc* is the average number of restricted nodes, i.e., label nodes and their ancestors. *Branching nodes/doc* - a branching node is a restricted node that has at least 2 child nodes. *Tokens/doc* - number of tokens per document after removing stopwords.

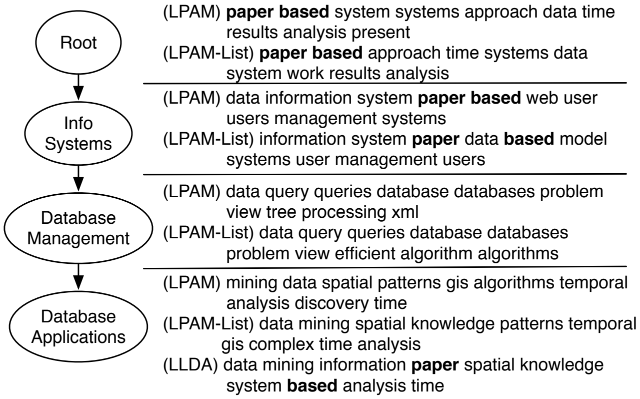

Root
(LPAM) **paper based** system systems approach data time results analysis present
(LPAM-List) **paper based** approach time systems data system work results analysis
Info Systems
(LPAM) data information system **paper based** web user users management systems
(LPAM-List) information system **paper** data **based** model systems user management users
Database Management
(LPAM) data query queries database databases problem view tree processing xml
(LPAM-List) data query queries database databases problem view efficient algorithm algorithms
Database Applications
(LPAM) mining data spatial patterns gis algorithms temporal analysis discovery time
(LPAM-List) data mining spatial knowledge patterns temporal gis complex time analysis
(LLDA) data mining information **paper** spatial knowledge system **based** analysis time

Figure 2: Four taxonomy nodes along with the corresponding top 10 words learned by LPAM, LPAM-List and Labeled LDA (LLDA). Note that LLDA provides no keywords for the internal nodes and that the new models successfully pull domain-specific words to the top of the hierarchy (see the examples shown in **bold**).

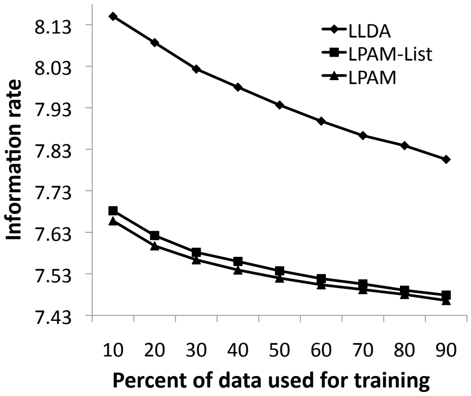


Figure 3: Information rate calculated on the ACM dataset. Note that lower is better. The difference between the models is statistically significant ($p < 0.001$).

Documents from each dataset are labeled with node(s) from a dataset-specific taxonomy. These taxonomies have tree structures. However, our models can be trivially extended to handle directed acyclic graphs as well. ACM documents are tagged with at least one leaf node. We ignore the additional descriptors beyond the leaf-node labels. If a document is tagged with a node called "General", that implies the paper is relevant to most of the nodes in that subtree. The models we propose and Labeled LDA require a specific set of document labels. For this reason we discard papers tagged with "General." These documents constitute only 4% of the dataset and our goal of attaining concept keywords can be achieved without them.

NYT articles can be tagged with multiple leaf or internal nodes. For 98% of the documents, an ancestor of a document label is also used to tag the article. However, in merely 0.6% of the documents all the ancestors are labels. Table 1 contains more statistics.

4. EXPERIMENTAL RESULTS

Let $\mathbf{1}^{(N)}$ be a vector of size N containing 1's. The parameters $\boldsymbol{\beta}$, $\boldsymbol{\alpha}^{(d)}$, and $\boldsymbol{\alpha}^{(d,c)}$ are set, respectively, to $0.01 \times \mathbf{1}^{(V)}$, $\mathbf{1}^{(|C_d|)}$, and $\mathbf{1}^{(K)}$, where V is the vocabulary size, C_d is the set of restricted nodes for document d, and K is the number of c's child nodes that are in C_d plus 1 (because of the *exit*).

4.1 Topic examples

Figure 2 shows the topics that LPAM and LPAM-List learn for a few ACM nodes—Database Applications and its ancestors. Labeled LDA considers only labels, so it produces topics only for leaf nodes such as Database Applications. We can see that the higher in the taxonomy the concepts are located, the more general the corresponding keywords are. The reason is that those concepts take part in the generation of many documents, so domain-specific stopwords (*e.g.*, "paper" and "based" highlighted in bold in Figure 2) common to all these documents are driven higher up the tree. Labeled LDA does not use internal nodes. As a result, general words are prominent in the distributions corresponding to the leaf nodes.

4.2 Information rate

We set aside 10% of the data for testing ($\mathbf{w}^{test}$) and 90% of the data for training ($\mathbf{w}^{train}$). We use the "left-to-right" algorithm [10] to compute $P(\mathbf{w_d}^{test}|\mathbf{w}^{train}, \mathbf{z}^{train}, \mathbf{U})$ where $\mathbf{U}$ is the set of hyperparameters and $\mathbf{w}_d^{test}$ denotes the words in the d'th test document. This probability is averaged across 5 runs to calculate the estimate $\tilde{P}_d$ and the information rate $-\frac{\ln \tilde{P}_d}{N_d}$, where N_d is the number of tokens in the test document.

Because of limited space, we present in Figure 3 the average information rate obtained using one of the datasets, the ACM corpus. We compare the models using a two-tailed paired t-test. LPAM consistently outperforms LPAM-List, which in turn is better than Labeled LDA. The differences between the models is statistically highly significant ($p < 0.001$) for varying amounts of training data. In the NYT

Action	Correct	Approx.	Incorrect
Addition	10	5	2
Deletion	21	-	5

Figure 4: User study results - actions taken after viewing discovered keywords

dataset LPAM achieves a slightly worse information rate compared to LPAM-List. However, both of these models outperform LLDA. As we point out in Section 3, if a NYT document is labeled with a given concept, then in many cases some of the ancestor nodes are also among the document tags. As a result, the margin between Labeled LDA and the other two models is smaller compared to the results we obtain with the ACM dataset. In future work, we plan to examine the differences between LPAM and LPAM-List.

4.3 Dataset navigation

We would like to know whether the concept keywords help taxonomy interpretation. We focus on the ACM dataset and use the keywords discovered by LPAM-List.

To address this question we designed and issued a survey to 15 graduate student volunteers at the Computer Science Department at University of Massachusetts Amherst. In our survey each participant is presented with a randomized list of abstracts and has to complete the following four tasks:
Step 1: Select four ACM abstracts that are about topics with which they are not significantly familiar. Since we have 15 participants, the total number of labeled abstracts is 60.
Step 2: For a given abstract, we present the user the subtrees rooted at the second level (the level under the top node) that contain the document's labels. In this way, we save the survey participants from needing to browse subtrees that are obviously irrelevant. The users are asked to label each abstract with at least one relevant tag among the leaves of the presented subtrees. The keywords that our models discovered are not presented.
Step 3: The users are shown the top 20 keywords associated with each concept and are asked to classify the same four abstracts they select in *Step 1*. If the users change any of the tags, they have to explain what prompted the change.
Step 4: The users comment whether they find it useful to have the LPAM-List's topic keywords at their disposal.

Note that although all participants are experts in computer science, they are not experts in the particular research sub-areas of the test documents. We analyze the node additions and deletions the participants made in *Step 3*. An addition can be *correct* (the added label is in the set of true labels), *approximate* (the added label shares a parent with a true label), or *incorrect* (none of the previous two cases). A deletion can be *correct* (the removed label is not in the set of true labels) or *incorrect* (the removed label is correct).

The results are summarized in Figure 4. The majority of the actions (36 out of 43) are beneficial. More specifically, the fraction of correctly removed labels is significantly greater than the fraction of incorrectly removed ones. This result indicates that keywords are helpful for identifying errors. Similarly, the fraction of correct and approximate additions is greater than the fraction of incorrect ones which leads us to the conclusion that concept keywords help users interpret taxonomies. Furthermore, in their responses in *Step 4* all but one of the surveyed students say that the keywords are helpful. We conclude that LPAM-List can play an important role in document classification.

5. CONCLUSIONS AND FUTURE WORK

We present two topic models that automatically build lists of clarifying keywords for each node in a given taxonomy. Our user study shows that presenting a few of the highest weighted keywords as concept summaries is beneficial for taxonomy interpretation. Also, the information rate produced by LPAM and LPAM-List is significantly better compared to Labeled LDA's performance.

In future work, we plan to extend LPAM, so that it can suggest edits to taxonomies. This is an important problem because new ideas emerge over time and the accompanying taxonomies should be adjusted accordingly. In fact, the output of LPAM was provided to the committee that edited the ACM Computing Classification System in 2011. We also plan to apply LPAM and LPAM-List to the problem of taxonomy alignment.

6. ACKNOWLEDGMENTS

The authors would like to thank Laura Dietz for useful discussions and ACM for providing metadata from their digital library. This work was supported in part by the Center for Intelligent Information Retrieval, in part by NIH award #HHSN271201000758P and in part by NSF grant #CNS-0958392.

7. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. In *JMLR*, 2003.

[2] D. M. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *NIPS*, 2003.

[3] J. Boyd-Graber and P. Resnik. Holistic sentiment analysis across languages: Multilingual supervised latent Dirichlet allocation. In *EMNLP*, 2010.

[4] W. Li, D. Blei, and A. McCallum. Nonparametric Bayes pachinko allocation. In *UAI*, 2007.

[5] D. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *ICML*, 2007.

[6] D. Newman, S. Karimi, and L. Cavedon. Using topic models to interpret MEDLINE's medical subject headings. In *Advances in Artificial Intelligence*, 2009.

[7] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP*, 2009.

[8] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI*, 2004.

[9] M. Steyvers, P. Smyth, and C. Chemudugunta. Combining background knowledge and learned topics. In *Topics in Cognitive Science*, 2010.

[10] H. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *ICML*, 2009.

[11] X. Xue, S. Huston, and W. B. Croft. Improving verbose queries using subset distribution. In *CIKM*, 2010.