ONEMercury: Towards Automatic Annotation of Environmental Science Metadata

Suppawong Tuarob[1], Line C. Pouchard[2], Natasha Noy[3], Jeffery S. Horsburgh[4], and Giri Palanisamy[2]

[1]Pennsylvania State University, University Park, PA, USA
[2]Oak Ridge National Laboratory, Oak Ridge, TN, USA
[3]Stanford University, Stanford, CA, USA
[4]Utah State University, Logan, UT, USA
suppawong@psu.edu, pouchardlc@ornl.gov, noy@stanford.edu, jeff.horsburgh@usu.edu, palanisamyg@ornl.gov

Abstract. The rapid growth of diverse data types and greater volumes available to environmental sciences prompts the scientists to seek knowledge in data from multiple places, times, and scales. To facilitate such need, ONEMercury has recently been implemented as part of the DataONE project to serve as a portal for accessing environmental and observational data across the globe. ONEMercury harvests metadata from the data hosted by multiple repositories and makes it searchable. However, harvested metadata records sometimes are poorly annotated or lacking meaningful keywords, and hence would unlikely be retrieved during the search process. In this paper, we develop an algorithm for automatic metadata annotation. We transform the problem into a tag recommendation problem, and propose a score propagation style algorithm for tag recommendation. Our experiments on four data sets of environmental science metadata records not only show great promises on the performance of our method, but also shed light on the different natures of the data sets.

1 Introduction

Environmental sciences have become both complex and data-intensive, needing accesses to heterogenous data collected from multiple places, times, and thematic scales. For example, research on climate changes would involve exploring and analyzing observational data such as the migration of animals and temperature shifts across the world, from time to time. While the needs to access such heterogenous data are apparent, the rapid expansion of observational data, in both quantity and heterogeneity, poses huge challenges for data seekers to obtain the right information for their research. Such problems behoove tools that automatically manage, discover, and link big data from diverse sources, and present the data in the forms that are easily accessible and comprehensible.

1.1 ONEMercury Search Service

Recently, DataONE, a federated data network built to facilitate accesses and preservation about environmental and ecological science data across the world, has come to exist and gain increasingly popularity[7]. DataONE harvests metadata from different environmental data providers and make it searchable via the search interface ONEMercury[1], built on Mercury[2], a distributed metadata man-

[1] https://cn.dataone.org/onemercury/
[2] http://mercury.ornl.gov/

agement system. ONEMercury offers two modes of searching: basic and advance. The basic mode only requires the user to input a set of keywords and the system would return matching results; while the advance mode adds the capability to further filter search results by authors, projects, and keywords.

1.2 Challenge and Proposed Solution

Linking data from heterogenous sources always has a cost. One of the biggest problems that ONEMercury is facing is the different levels of annotation in the harvested metadata records. Poorly annotated metadata records tend to be missed during the search process as they lack meaningful keywords. Furthermore, such records would not be compatible with the advance mode offered by ONEMercury as it requires the metadata records be semantically annotated with keywords from the keyword library. The explosion of the amount of metadata records harvested from an increasingly number of data repositories makes it even impossible to annotate the harvested records manually by hand, urging the need for a tool capable of automatically annotating poorly curated metadata records.

In this paper, we address the problem of automatic annotation of metadata records. Our goal is to build a fast and robust system that annotates a given metadata record with meaningful and related keywords from a given ontology. The idea is to annotate a poorly annotated record with keywords associated to the well annotated records that it is most similar with. We propose a solution to this problem by first transforming the problem into a tag recommendation problem where the set of recommended tags is used to annotate the given metadata record, and then propose an algorithm that deals with the problem.

1.3 Problem Definition

We define a document as a tuple of textual content and a set of tags. That is `d = <c, e>`, where `c` is the textual content, represented by a sequence of terms, of the document `d` and `e` is a set of tags associated with the document. Given a tag library T, a set of annotated documents S, and a non-annotated query document q, our task is to recommend a ranked set of K tags taken from T to the query q. A document is said to be annotated if it has at least one tag; otherwise, it is non-annotated. The formal description of each variable is given below:

$$T = \{t_1, t_2, ..., t_M\}\,; t_i \ is\ a\ tag. \tag{1}$$

$$S = \{s_1, s_2, ..., s_N\}\,; s_i = \langle c_{si}, e_{si}\rangle\,, e_{si} \subseteq T, and\ e_{si} \neq \emptyset \tag{2}$$

$$q = \langle c_q, \emptyset\rangle \tag{3}$$

1.4 Contributions

This paper has four key contributions as follows:

1. We address a real word problem of linking data from multiple archives faced by ONEMercury. We transform the problem into the tag recommendation problem, and generalize the problem so that the proposed solution can apply to other domains.
2. We propose a novel score propagation technique for tag recommendation. Given a document query `q`, we first calculate the similarity score between the query and each document in the source `S`. The score then is propagated to the tags of each document in the source. Tags then are ranked by the

scores, and the top K tags are returned for recommendation. We propose two different measures for computing the similarity between two documents: term frequency-inverse document frequency (TFIDF) and topic model (TM).
3. We crawl environmental science metadata records from 4 different archives for our data sets: the Oak Ridge National Laboratory Distributed Active Archive Center (DAAC)[3], Dryad Digital Repository[4], the Knowledge Network for Biocomplexity (KNB)[5], and TreeBASE: a repository of phylogenetic information[6]. We select roughly 1,000 records from each archive for the experiments.
4. We validate our proposed method using aggressive empirical evaluations. We use document wise 10 fold cross validation to evaluate our schemes with 5 evaluation metrics: Precision, Recall, F1, MRR (Mean Reciprocal Rank), and BPref (Binary Preference). These evaluation metrics are extensively used together to evaluate recommendation systems.

2 Related Works

Since we choose to transform our setting to a tag recommendation problem. We briefly state the related literature here. Tag recommendation has gained substantial amount of interest in recent years. Most work, however, focuses on personalized tag recommendation, suggesting tags to a user's object based on the user's preference and social connection. Mishne et al. [8] employ the social connection of the users to recommend tags for weblogs, based on similar weblogs tagged by the same users. Wu et al.[10] utilize the social network and the similarity between the contents of objects to learn a model for recommending tags. Their system aims towards recommending tags for Flickr photo objects. While such personalized schemes have been proven to be useful, some domains of data have limited information about authors (users) and their social connections. Liu et al. [6] propose a tag recommendation model using Machine Translation. Their algorithm basically trains the translation model to translate the textual description of a document in the training set into its tags. Krestel et al.[5] employ topic modeling for recommending tags. They use the Latent Dirichlet Allocation algorithm to mine topics in the training corpus, using tags to represent the textual content. They evaluate their method against the association rule based method proposed in [4].

3 Data Sets

We obtain the data sets of environmental metadata records for the experiments from 4 different archives: the Oak Ridge National Laboratory Distributed Active Archive Center (DAAC), Dryad Digital Repository (DRYAD), the Knowledge Network for Biocomplexity (KNB), and TreeBASE: a repository of phylogenetic information (TREEBASE). The statistics of the data sets including the number

[3] http://daac.ornl.gov/
[4] http://datadryad.org/
[5] http://knb.ecoinformatics.org/index.jsp
[6] http://treebase.org/treebase-web/home.html

	# Docs	#All Tags	Avg Tags/Doc	#Uniq. Tags	Tag Util.	#All Words	Avg Words/Doc
DAAC	978	7,294	7.46	611	11.937	101968	104.261
DRYAD	1,729	8,266	4.78	3,122	2.647	224,643	129.926
KNB	24,249	254,525	10.49	7,375	34.511	1535560	63.324
TREEBASE	2635	1838	0.697	1321	1.391	30054	11.405

Table 1: Statistics of the 4 data sets.

of documents, total number of tags, average number of tags per document, number of unique tags (tag library size), tag utilization, number of all words (data set size), and average number of word per document, are summarized in Table 1. Tag utilization is the average number of documents where a tag appears in, and is defined as $\frac{\#\ all\ tags}{\#\ unique\ tags}$.

In our setting, we assume that the documents are independently annotated, so that the tags in our training sets represent the gold-standard. However, some metadata records may not be independent since they may be originated from the same projects or authors, hence annotated with similar styles and sets of keywords. To mitigate such problem, we randomly select a subset of 1,000 annotated documents (except DAAC data set, which only has 978 documents, hence we select them all.) from each archive for our experiments. We combine all the textual attributes (i.e. `Title`, `Abtract`, `Description`) together as the textual content for the document. We preprocess the textual content in each document by removing 664 common stop words and punctuation, and stemming the words using the Porter2 stemming algorithm.

4 Preliminaries

Our proposed solution is built upon the concepts of Cosine Similarity, Term Frequency-Inverse Document Frequency (TFIDF), and Latent Dirichlet Allocation (LDA). We briefly introduce them here to fortify readers' background before going further.

4.1 Cosine Similarity

In general, cosine similarity is a measure of similarity between two vectors by measuring the cosine of the angle between them. Given two vectors A and B, the cosine similarity is defined using a dot product and magnitude as:

$$CosineSim(A, B) = \frac{A \cdot B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{N} A_i \times B_i}{\sqrt{\sum_{i=1}^{N} (A_i)^2} \times \sqrt{\sum_{i=1}^{N} (B_i)^2}} \tag{4}$$

`CosineSim(A,B)` outputs `[0,1]`, with 0 indicating independence, and the value in between indicates the level of similarity. The cosine similarity is heavily used to calculated the similarity between two vectorized documents.

4.2 Term Frequency-Inverse Document Frequency

TF-IDF is used extensively in the information retrieval area. It reflects how important a term is to a document in a corpus. TF-IDF has two components: the term frequency (TF) and the inverse document frequency (IDF). The TF is the frequency of a term appearing in a document. The IDF of a term measures how important the term is to the corpus, and is computed based on the document frequency, the number of documents in which the term appears. Formally, given

a term t, a document d, and a corpus (document collection) D:

$$tf(t,d) = \sqrt{count(t,d)},\ idf(t,D) = \sqrt{log\left(\frac{|D|}{|d \in D; t \in d|}\right)},\ tfidf(t,d,D) = TF(t,d) \cdot IDF(t,D) \quad (5)$$

We can then construct a TF-IDF vector for a document `d` given a corpus `D` as follows:

$$TFIDF(d,D) =< tfidf(t_1,d,D), tfidf(t_2,d,D), \cdots, tfidf(t_n,d,D) > \quad (6)$$

Consequently, if one wishes to compute the similarity score between two documents d_1 and d_2, the cosine similarity can be computed between the TF-IDF vectors representing the two documents:

$$DocSim_{TFIDF}(d_1,d_2,D) = CosineSim\left(TFIDF(d_1,D), TFIDF(d_2,D)\right) \quad (7)$$

4.3 Latent Dirichlet Allocation

In text mining, the Latent Dirichlet Allocation (LDA) [2] is a generative model that allows a document to be represented by a mixture of topics. The basic intuition of LDA for topic modeling is that an author has a set of topics in mind when writing a document. A topic is defined as a distribution of terms. The author then chooses a set of terms from the topics to compose the document. With such assumption, the whole document can be represented using a mixture of different topics. Mathematically, the LDA model is described as follows:

$$P\left(t_i|d\right) = \sum_{j=1}^{|Z|} P\left(t_i|z_i = j\right) \cdot P(z_i = j|d) \quad (8)$$

$P(t_i|d)$ is the probability of term t_i being in document d. z_i is the latent (hidden) topic. $|Z|$ is the number of all topics. This number needs to be predefined. $P(t_i|z_i = j)$ is the probability of term t_i being in topic j. $P(z_i = j|d)$ is the probability of picking a term from topic j in the document d.

After the topics are modeled, we can assign a distribution of topics to a given document using a technique called *inference*. A document then can be represented with a vector of numbers, each of which represents the probability of the document belonging to a topic.

$$Infer(d,Z) =< z_1, z_2, ..., z_Q >; |Z| = Q$$

Where Z is a set of topics, d is a document, and z_i is a probability of the document d falling into topic i. Since a document can be represented using a vector of numbers, one can then compute the topic similarity between two documents d_1 and d_2 using cosine similarity as follows:

$$DocSim_{TM}(d_1,d_2,Z) = CosineSim\left(Infer(d_1,Z), Infer(d_2,Z)\right) \quad (9)$$

5 Method

In this section, we describe our score propagation method for tag recommendation. We show how our algorithm works using a simple example in Section 5.1, and later discuss the variation of document similarity measures that we use.

5.1 System Overview

Figure 1 illustrates a flow of our score propagation algorithm on a simple example. Three documents in the source are annotated with tags {`water`, `seagull`}, {`seagull`, `soil`, `bird`}, and {`bird`, `air`} respectively. Our algorithm proceeds as follows:

STEP1 The document similarity score is computed between the document query

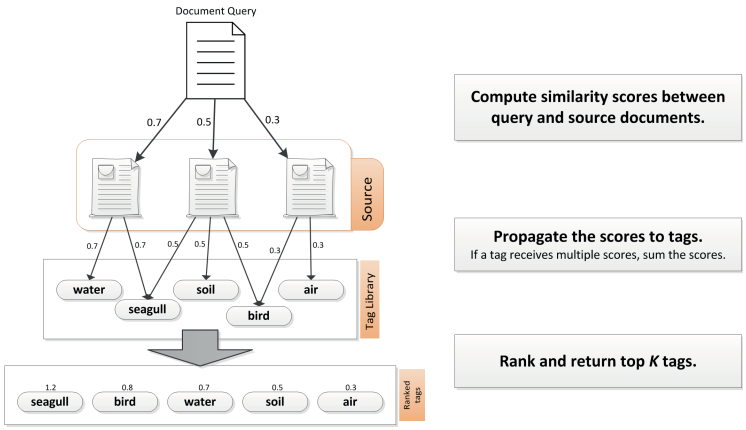


Fig. 1: Overview of the score propagation method.

and each document in the source.
STEP2 The scores then are propagated to the tags in each source document.The scores are combined if a tag receives multiple scores. In the example, tags `seagull` and `bird` obtain multiple scores (0.7+0.5) and (0.5+0.3) respectively.
STEP3 The tags are ranked by the scores. Then the top `K` tags are returned as suggested tags.

5.2 Document Similarity Measures

We explore two different document similarity measures when computing the similarity between the document query and the documents in the source.

TFIDF Based. The first measure relies on the term frequency-inverse document frequency discussed in Section 4.2. In our setting, D is the document source. In order to compute the IDF part of the scheme, all the documents in the source need to first be indexed. Hence the training phase (preprocess) involves indexing all the documents. We then compute the similarity between the query q and a source document d using $DocSim_{TFIDF}(q, d, D)$ as defined in Equation 7. We use LingPipe[7] to perform the indexing and calculating the TFIDF based similarity.

TM Based. The second document similarity measure utilizes topic distributions of the documents. Hence the training process involves modeling topics from the source using LDA algorithm as discussed in Section 4.3. We use Stanford Topic Modeling Toolbox[8] with the collapsed variational Bayes approximation[1] to identify topics in the source documents. For each document we generate unigrams, bi-grams, and tri-grams, and combine them to represent the textual content of the document. The algorithm takes two input parameters: the number of topics to be identified and the maximum number of the training iterations. After some experiments on varying the two parameters we fix them at 300 and

[7] http://alias-i.com/lingpipe/
[8] http://nlp.stanford.edu/software/tmt/tmt-0.4/

1,000 respectively. For assigning a topic distribution to a document, we use the inference method proposed by [1].

6 Evaluation and Discussion

We evaluate our methods using the tag prediction protocol. We artificially create a test query document by removing the tags from an annotated document. Our task is to predict the removed tags. There are two reasons behind the choosing of this evaluation scheme:

1. The evaluation can be done fully automatically. Since our data sets are large, manual evaluation (i.e. having human identify whether a recommended tag is relevant or not) would be infeasible.
2. The evaluation can be done against the existing gold standard established (manually tagged) by expert annotators who have good understanding about the data, while manual evaluation could lead to evaluation biases.

We test our algorithm using with document similarity measures on each data set, using two different source (training set) modes: self-source and cross-source. For the self-source mode, the documents in the training set are selected from the same archive as the query; while the cross-source mode combines the training documents from all the archives together. We evaluate our algorithms with different source modes using document-wise 10 fold cross validation, where each data set is split into 10 equal subsets, and for each fold $i \in \{1, 2, 3, ..., 10\}$ the subset i is used for the testing set, and the other 9 subsets are combined and used as the source (training set). The results of each fold are summed up and the averages are reported. The evaluation is done on a Windows 7 PC with Intel Core i7 2600 CPU 3.4 GHz and 16GB of ram.

6.1 Evaluation Metrics

Precision, Recall, F1.
For each document query in the test set, we use the original set of tags as the ground truth T_g. Assume that the set of recommended tags are T_r , so that the correctly recommended tags are $T_g \bigcap T_r$. Precision, recall and F1 measures are defined as follows:

$$precision = \frac{|T_g \bigcap T_r|}{|T_r|}, recall = \frac{|T_g \bigcap T_r|}{|T_g|}, F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

In our experiments, the number of recommended tags ranges from 1 to 30. It is wise to note that better tag recommendation systems tend to rank correct tags higher than the incorrect ones. However, the precision, recall, and F1 measures do not take ranking into account. To evaluate the performance of the ranked results, we employ the following evaluation metrics.

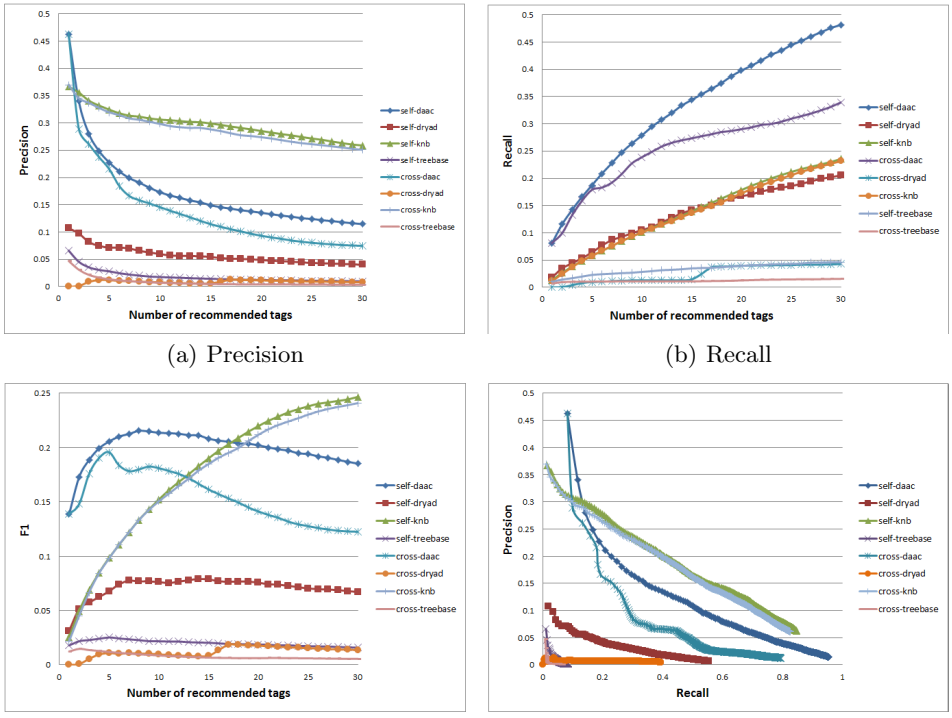
Mean Reciprocal Rank (MRR).
MRR measure takes ordering into account. It measures how well the first correctly recommended tag is ranked. Formally, given a testing set Q, let $rank_q$ be the rank of the first corrected answer of query $q \in Q$, then MRR of the query set Q is defined as follows:

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{rank_q}$$

Binary Preference (Bpref).
Bpref measure considers the order of each correctly recommended tag [3]. Let S be the set of recommended tags by the system, R be the set of corrected tags , $r \in R$ be a correct recommendation, and $i \in S - R$ be an incorrect recommendation. The Bpref is defined as follows:

$$Bpref = \frac{1}{|R|} \sum_{r \in R} 1 - \frac{|i \ ranked \ higher \ than \ r|}{|S|}$$

6.2 Evaluation on TFIDF-Based Method



(a) Precision (b) Recall

(c) F1 (d) Precision vs Recall

Fig. 2: Precision, Recall, F1, and Precision-vs-Recall of the TFIDF based method performed on different data sets and source selection modes.

We run our algorithm with TFIDF based document similarity on each of the 4 data sets, using both self and cross source modes. Figure 2 summarizes the precision, recall, F1, and the prec. vs rec. graph. Table 3 summarizes the MMR, Bpref, and other time-wise (training and recommending) performances. *TTT*, *TRT*, *ATT*, and *ART* stand for *Total Train Time*, *Total Recommend Time*, *Average Train Time (per fold)*, and *Average Recommend Time (per fold)* respectively.

Metric	Self-recommendation				Cross-recommendation			
	daac	dryad	knb	treebase	daac	dryad	knb	treebase
MRR	0.754	0.326	0.922	0.074	0.703	0.298	0.887	0.070
Bpref	0.900	0.493	0.909	0.063	0.868	0.468	0.896	0.064
TTT (hours)	6.752	12.461	3.221	1.115	24.127	24.127	24.127	24.127
TRT (min)	8.270	13.960	8.266	2.831	32.621	33.446	32.906	32.598
ATT (sec)	2430.822	4486.089	1159.811	401.497	8686.002	8686.002	8686.002	8686.002
ART (sec)	49.625	83.764	49.597	16.99	195.73	200.68	197.44	195.59

Table 2: MRR, Bpref, and other time-wise performance statistics of the **TM** based method performed on different data sets and source selection modes.

Metric	Self-recommendation				Cross-recommendation			
	daac	dryad	knb	treebase	daac	dryad	knb	treebase
MRR	0.564	0.202	0.494	0.089	0.532	0.032	0.514	0.058
Bpref	0.818	0.440	0.665	0.069	0.630	0.245	0.648	0.044
TTT (sec)	52.937	63.23	60.634	61.5	69.322	69.251	69.176	69.221
TRT (sec)	9.946	11.940	12.295	10.752	43.817	44.963	44.478	43.581
ATT (sec)	5.293	6.323	6.063	6.15	6.932	6.925	6.917	6.922
ART (sec)	0.994	1.194	1.229	1.075	4.381	4.496	4.447	4.358

Table 3: MRR, Bpref, and other time-wise performance statistics of the **TFIDF** based method performed on different data sets and source selection modes.

From the results, we get the following observations. First, the performance differs significantly across data sets. Overall, the TFIDF based method performs better on DAAC and KNB data sets. DAAC data set has smaller tag library (only 611 unique tags), hence the chance of recommending correct tags (which is reflected by the recall growth rates) is higher than those of other data sets. The KNB data set, though has the largest tag library, has a high tag utilization rate, hence the chance of correctly guessing the tags is expectedly higher.

Second, self-source recommendations always perform better than cross-source recommendations with respect to our evaluation scheme. This is because, given a document query, the cross-recommendation system may introduce alien tags from other data sets, which would most certainly be identified as incorrect tags. Note that, even though the cross-recommendation systems may perform worse than the self-recommendation ones with respect to our evaluation setting, in real world these alien tags may actually have semantic relevance to the query.

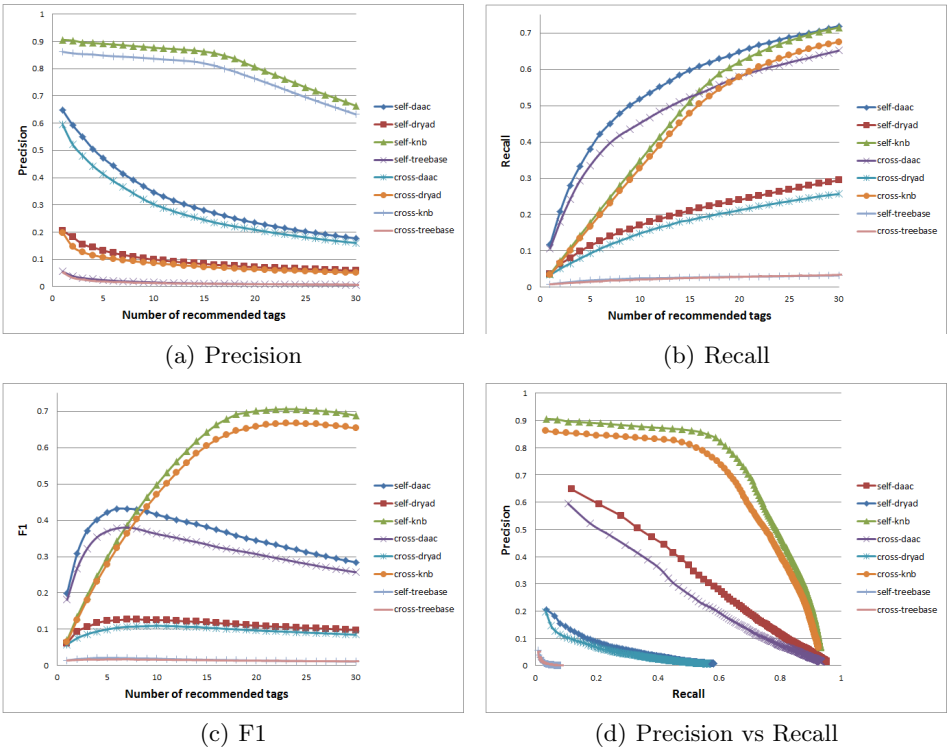
6.3 Evaluation on TM-Based Method

We run our score propagation algorithm with the *TM* based document similarity on each data set, using both self and cross recommendation modes. Figure 3 summarizes the precision, recall, F1, along with the prec. vs rec. comparison. Table 3 summarizes the MMR, Bpref, and other time-wise performances.

The comparison among different data sets is similar to the result from the TFIDF based method, except that the performance on the KNB data set seems to be surprisingly outstanding.

6.4 Performance Comparison Between the Methods

We compare the performances of the two document similarity measures (TFIDF based and TM based) on the 4 data sets with self-recommendation. Figure 4



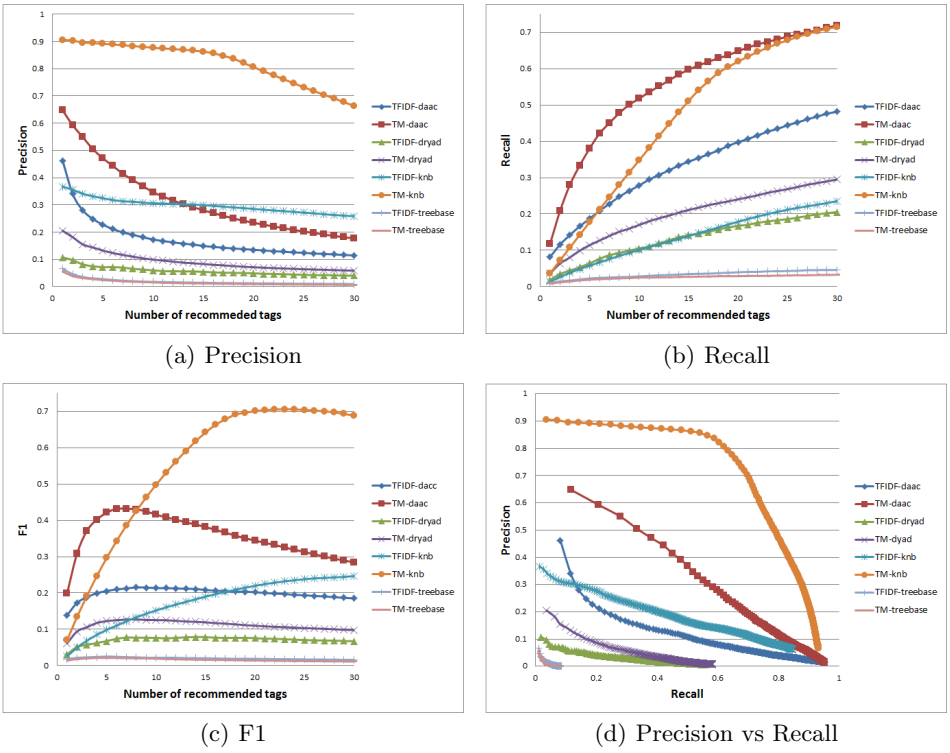
(a) Precision (b) Recall

(c) F1 (d) Precision vs Recall

Fig. 3: Precision, Recall, F1, and Precision-vs-Recall of the TM based method performed on different data sets and source selection modes.

summarizes the precision, recall, F1, and prec. vs rec. graphs. The TM based approach obviously outperforms the TFIDF based approach in DAAC, DRYAD, and KNB data sets. The performance of the TM based approach is dominant when applied to the KNB data set, as seen in the precision vs recall graph in Figure 4(d), where the curve forms the shape close to the ideal precision-vs-recall curve. The comparison, however, is not dominant on the TreeBASE data set. Actually, both algorithms perform very poorly on the TreeBASE data set. We hypothesize that this is because the TreeBASE documents are very sparse and have very few tags. From our statistics, each document in the TreeBASE data set has only 11 words and only 0.7 tags on average. Such sparse texts lead to weak relationship when finding textually similar documents in the TFIDF based approach, and the poor quality of the topic model used by the TM based approach. The small number of tags per document makes it even harder to predict the right tags.

We note that, though overall the TM based approach recommends better quality of tags, the training times take significantly longer than those of the TFIDF based approach. For example, it takes roughly 33 minutes to train the



(a) Precision (b) Recall

(c) F1 (d) Precision vs Recall

Fig. 4: Comparison between the TFIDF and TM approaches on different data sets, using self-sources.

TM based method (modeling topics) using 3,600 documents, while it takes only 7 seconds to train (index) the same amount of the documents via the TFIDF based approach. However, also note that the evaluation is done on a local PC. The issue of training times could be much diminished if the system is employed on a powerful computing server.

7 Conclusions and Future Research

In this paper we propose an algorithm for automatic metadata annotation. We are inspired by the real world problem faced by ONEMercury, a search system for environmental science metadata harvested from multiple data archives, in which the metadata from different archives has different levels of curation and hence behooves the system that automatically annotates poorly annotated metadata records. We treat each metadata record as a tagged document, and then transform the problem into a tag recommendation problem.

We propose the score propagation model for tag recommendation, with two variations of document similarity measures: TFIDF based and Topic Model (TM) based. The TM based approach yields impressive results, though with a cost of longer training times.

Our future works include evaluating our approaches against a well known state-of-the-art such as the method that mines association rules for tag recommendation[4]. We also plan to adopt a classification technique such as [9] to rank tags in the tag library. Finally, we aim to implement our automatic metadata annotation system into ONEMercury search service. This would give rise to further implementation and system integration issues.

References

1. Asuncion, A., Welling, M., Smyth, P., Teh, Y.W.: On smoothing and inference for topic models. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. pp. 27–34. UAI '09, AUAI Press, Arlington, Virginia, United States (2009), http://dl.acm.org/citation.cfm?id=1795114.1795118
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (Mar 2003), http://dl.acm.org/citation.cfm?id=944919.944937
3. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 25–32. SIGIR '04, ACM, New York, NY, USA (2004), http://doi.acm.org/10.1145/1008992.1009000
4. Heymann, P., Ramage, D., Garcia-Molina, H.: Social tag prediction. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 531–538. SIGIR '08, ACM, New York, NY, USA (2008), http://doi.acm.org/10.1145/1390334.1390425
5. Krestel, R., Fankhauser, P., Nejdl, W.: Latent dirichlet allocation for tag recommendation. In: Proceedings of the third ACM conference on Recommender systems. pp. 61–68. RecSys '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1639714.1639726
6. Liu, Z., Chen, X., Sun, M.: A simple word trigger method for social tag suggestion. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 1577–1588. EMNLP '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011), http://dl.acm.org/citation.cfm?id=2145432.2145601
7. Michener, W., Vieglais, D., Vision, T., Kunze, J., Cruse, P., Janée, G.: DataONE: Data Observation Network for Earth - Preserving Data and Enabling Innovation in the Biological and Environmental Sciences. DLib Magazine 17(1/2), 1–12 (2011), http://www.dlib.org/dlib/january11/michener/01michener.html
8. Mishne, G.: Autotag: a collaborative approach to automated tag assignment for weblog posts. In: Proceedings of the 15th international conference on World Wide Web. pp. 953–954. WWW '06, ACM, New York, NY, USA (2006), http://doi.acm.org/10.1145/1135777.1135961
9. Treeratpituk, P., Teregowda, P., Huang, J., Giles, C.L.: Seerlab: A system for extracting key phrases from scholarly documents. In: Proceedings of the 5th International Workshop on Semantic Evaluation. pp. 182–185. SemEval '10, Association for Computational Linguistics, Stroudsburg, PA, USA (2010), http://dl.acm.org/citation.cfm?id=1859664.1859703
10. Wu, L., Yang, L., Yu, N., Hua, X.S.: Learning to tag. In: Proceedings of the 18th international conference on World wide web. pp. 361–370. WWW '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1526709.1526758