Automatic Class Labeling for CiteSeer^X

Surya Dhairya Kashireddy
Susan Gauch
Syed Masum Billah
Computer Science and Computer Engineering
University of Arkansas
Fayetteville, AR 72701

Abstract— **The CiteSeer^x project at the University of Arkansas uses a browsing interface is based on the Association for Computing Machinery's Computing Classification System (ACM CCS). CCS contains just 369 categories whereas the CiteSeerx database contains over 2 million documents. This results in more than 6500 documents per category, far too many to browse. To address this problem, we are exploring ways to automatically expand the CCS ontology. Previous work has focused on using clustering to automatically identify the new clas-ses. This work focuses on how to label the subclasses in a semantically meaningful way to that they can sup-port user browsing. We develop methods based on text mining from the subclass members to extract class la-bels. We evaluate three methods by comparing the suggested labels with human-assigned labels for exist-ing categories.**

Keywords-ontologies, labeling, text mining

I. INTRODUCTION

With the maturity of the Internet and improvements in storage technologies, the Web is overwhelmed with data. As a result, browsing system based interfaces need to employ new approaches that support users exploring massive amounts of data. Many browsing systems are based on an ontology or conceptual hierarchy that represents the underlying knowledge domain. By exploring classes within the ontology, users can browse information in an organized manner.

As the number of documents in a corpus/collection increases, the number of classes to which they are related also increases. Therefore, if an ontology remains static as documents are added, then too many documents are associated with each class, affecting the scalability and usability of the browsing interface. Ontologies need to grow as the information they organize grows. Traditionally, growing and maintaining an ontology is a manual, labor-intensive task that requires formal ontology engineers familiar with domain-specific knowledge. Research into automatic ontology expansion focuses on providing more scalable techniques to provide ontology adaptation.

One approach to ontology expansion divides the contents of a class in an existing ontology into classes using clustering techniques. This is automatic division of leaf classes into subclasses such that data points in the same subclass are closely related to each other and also far apart from data points in other subclasses. Researchers have developed various algorithms to perform this division, typically employing clustering techniques to identify and form the new subclasses.

Although clustering techniques for text documents have been widely studied, less attention has been paid to creating good cluster descriptors. Often times, automatically created cluster descriptors either fail to provide a comprehensive description of the cluster or consist of lists of terms from which a person cannot infer a general description.

In this paper, we propose a new algorithm that automatically assigns concise labels to subclasses in a hierarchical ontology. Our contributions are as follows: (1) we describe an automatic ontology expansion technique based on clustering; and (2) we evaluate a new labeling algorithm to label the newly created subclasses.

The rest of the paper is organized as follows. Section 2 presents previous research on ontologies and cluster labeling. Section 3 describes the architecture and the ontology expansion process in detail. Section 4 provides experimental results. Section 5 summarizes our findings and offers suggestions about possible future improvements.

II. RELATED WORK

A. Ontologies

The new era of the Semantic Web [22] has enabled users to extract semantically relevant data from the web [12]. The Semantic Web relies heavily on formal ontologies to structure data for comprehensive and transportable machine understanding. For the Semantic Web to be successful, we will need wide-scale availability of ontologies across many domains [15]. In practice, building ontology for intelligent systems involves domain-specific experts' efforts to manually identify a set of representational primitives [19] and integrate them into an ontology system [14,19,20].

B. Ontology Creation/Refinement

To address the ontology creation bottleneck, researchers are exploring ways to reuse ontologies and to build them using semi-automatic and automatic techniques [20, 21]. Velardi et al. [1] give a comprehensive overview of approaches for constructing ontologies. They also introduce a new semi-automatic technique to build domain ontologies. Speretta et al. [2] present an automatic technique, which selects appropriate domain ontologies from the collection of already existing ontologies of a given set of documents. Carmel et al. [7] suggests, selecting appropriate domain ontology and refine the ontologies based on document collection.

In [6], Carvalheria et al. provide a method for the semi-automatic construction of ontologies using texts of any domain for the extraction of classes and relations. By comparing the relative frequency of terms in the text with their typical, expected use, the method identifies classes and relations; they then represent the corresponding ontology using OWL so that it can be used by other applications.

Researchers working on the creation or expansion of ontologies need an evaluation technique to measure the quality of the created ontology. According to Fang et al. [9], the quality of formal ontology requires both a good conceptualization of a domain and a good specification of the conceptualization. They suggest that the terminology and the structure of the ontologies also need to be evaluated.

C. Cluster Labeling

Cluster labeling is a process of extracting unique features or descriptors to represent a cluster of topically related documents. [13, 17] review the performance of a variety of automatic labeling methods. Their results show that differential labeling outperforms cluster-internal labeling and hybrid methods.

The most common cluster descriptors are either concise labels or lists of terms and phrases. For example, Geraci et al. [3] describe ARMIL, a meta-search engine that groups into unique labeled clusters. Their cluster generation is done by the furthest-point-first algorithm, an approximation of k-center clustering, and the labeling is considered as a combination of intra-cluster and inter-cluster term extraction.

Carmel et al. [7] also describes a general framework for cluster labeling that extracts cluster labels from Wikipedia. They suggest extracting most important keywords or bigrams from clusters that are then presented as queries to search engine to retrieve relevant Wiki pages. The resulting topics or related terms associated with those Wiki pages are selected as labels for the clusters. Similarly, a machine-readable dictionary that is organized conceptually can also be used for labeling [14].

III. ONTOLOGY EXPANSION

In this section we discuss our ontology expansion approach in detail. As many researchers suggest, we start with an appropriate domain ontology relevant to our document collection in CiteSeer^X. We choose ACM's *Computing Classification System (CCS)* [10], a 3-level deep ontology that classifies scientific papers; then we will expand only those *CCS* classes that are growing rapidly and/or contain thousands of documents. For example, *Network Protocols (C.2.2)* is a class in the *CCS*, that has been extensively studied and it contains papers describing over 70 protocols for the various layers of the OSI model. It continues to grow dynamically as more and more documents are published. The current CiteSeer^x corpus contains more than 2762 documents related to this topic. Browsing through all of the documents in this class to locate those relevant to a particular protocol would be a time consuming task for users. It, and other classes with many associated documents, need to be split into subclasses for effective browsing, partitioning the documents into smaller, more cohesive clusters.

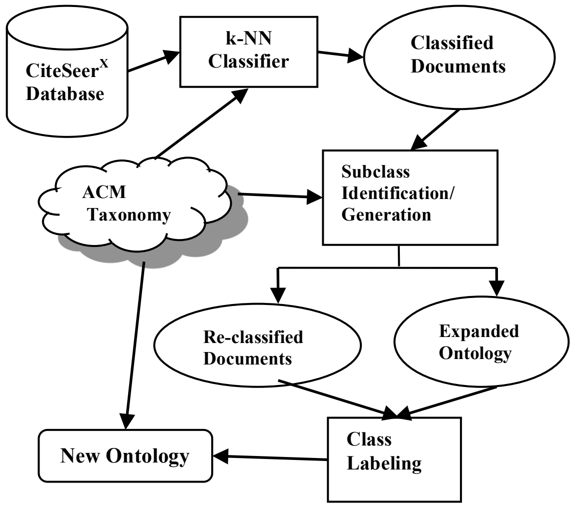

Fig 1. Ontology Expansion Model

A. Document classification

Only 2.6% percent of the CiteSeer^X corpus is tagged with classes from ACM's CCS. To associate the rest of the documents with CCS classes, we used a k-nearest-neighbors (kNN) classifier [16] previously built as part of the KeyConcept [5] project. The author-tagged documents were collected and used as training documents for the classifier. The untagged documents were then compared to the training documents based on their vocabulary and classified into the best matching class.

As discussed earlier, the large number of published documents available means that too many documents are classified into the same class, making the existing CCS unsuitable for a browsing interface for CiteSeer^X. Therefore, we need to divide large classes into sub-classes to enhance browsing system. Based on the results of Nanhong et al. [4], we use the *Direct* clustering algorithm implemented in *Cluto* [11, 18] to expand the existing 3-level deep *CCS* ontology into a 4-level deep ontology. In contrast to the repeated bisection method, the k-clustering solution is computed by simultaneously finding all k-clusters. The results of [4] on CiteSeer^X collections shows *"Direct algorithm"* has the minimum entropy and maximum purity compared to other algorithms in *Cluto*.

B. Labeling Subclasses

After clustering, the next task is to append these clusters as subclasses in ontology, requiring labels to appropriately describe the documents classified into them. Hence, our primary goal is to assign concise labels to subclasses as if they were manually assigned. As a first step to this process, we perform part-of-speech tagging on the document content, allowing us to focus our text mining for labels on nouns, the most common part of speech used for class labels.

There are two primarily features considered by our class label text miner: the frequency of the word within a cluster and distribution of word occurrences across the sibling clusters. In the next section, we discuss three different label-weighting methods we developed and evaluated to label subclasses.

C. Label-weighting Methods

We present three label-weighting methods, our baseline and two novel methods that we will evaluate in the next section.

TFIDF$_{ic}$:

This is one of the classic formulae commonly used by the information retrieval community. TFIDF stands for term frequency–inverse document frequency, a measure that calculates the importance of a word as a descriptor of document contents based the word's frequency within the document and its frequency of occurrence in all documents in the corpus. Treeratpituk et al. [8] calculates $TFIDF_{ic}$ as follows:

$$TFIDF_{ic} = TF_{ic} * \log\left(\frac{NC}{DF_i}\right)$$

where TF_{ic}, is the frequency of word i in cluster c, DF_i represents the number of clusters in which word i appears, and *NC* is the total number of clusters. The log of *NC/DF$_i$* is the classic inverse document frequency measure. A word with high TFIDF value is expected to be more important to the cluster and thus be a good cluster descriptor. Because of its widespread use, we chose TFIDF as our baseline for label selection.

DeltaTF$_{ic}$:

This method is our first novel method. It simply identifies terms that appear much more frequently in the current cluster as compared to other clusters. More formally, *DeltaTF$_{ic}$* is a frequency of term *i* in cluster *c* calculated as minus the average term frequency of *i* in rest of the sibling clusters. Computationally this is very inexpensive and quite scalable.

$$DeltaTF_{ic} = TF_{ic} - \frac{\sum_{c' \in \{C-c\}} TF_{ic'}}{NC-1}$$

where TF_{ic}, is the frequency of word i in cluster c, and *NC* is the total number of clusters

TFStDev$_{ic}$:

This is a novel approach uses the same TF metric as the first method but, rather than using inverse document frequency to capture the term distribution across clusters, it uses standard deviation. Standard deviation is a widely used statistical metric that measures the variation of a particular data point from the sample mean. In labeling subclasses, this metric is applied to locate words that occur in one cluster more frequently than would be statistically expected. These statistically surprising words are then selected as class labels.

$$TFStDev_{ic} = TF_{ic} * Standard\ Deviation\ of\ RTF_i$$

where relative term frequency, $RTF_i = \frac{\mathrm{TF_{ic}}}{|c|}$, the term frequency in cluster *c,* normalized by the cluster size |c|, the number of term occurrences within the cluster. Note, we also evaluated using the standard deviation of TF in this formula, but it did not perform as well, so it is not discussed further.

EVALUATION

The previous section provides automated ontology expansion technique and labeling methods. Now, we carry out our experiments to examine how each of these methods performs in ontology expansion and labeling.

A. Data Set:

As mentioned earlier, we are using ACM's *CCS* ontology for our CiteSeer^X collection. *CCS* contains 369 classes organized in a 3-level deep hierarchy. Operationally, we want to label the *level-4* clusters produced by *Cluto*. However, in order to evaluate our labeling algorithms, we need to compare the labeling results in clusters whose true names are known. Thus, we select 20 *level-2* classes and select all the human classified *level-3* subclasses of these classes. There were a total of 150 level-3 classes in our test set.

We compare our automatically extracted labels with those manually assigned as part of the CCS to see how well our text mining approach matches the true class names. For each selected class, we randomly choose 100 documents and generated labels for them using our aforementioned algorithms.

B. *Labeling Evaluation:*

To evaluate labeling of clusters, we need to compare the results produced with human assigned titles. We consider the highest-weighted 3 labels produced by different algorithms. Each of these labels is compared with human assigned titles. We evaluate the quality of labeling method with the F-measure described in [4] that combines *precision* and *recall* together into a single metric. *Precision* calculates the percentage of extracted labels that are the same as the true human-assigned labels. It essentially measures the accuracy of the extracted labels.

$$Precision = \frac{\sum_{i=1}^{N} M_i}{N}$$

where $M_i = 1$ if the i^{th} label occurs within the actual titles. $M_i = 0$ if the i^{th} label does not match with user assigned titles, and N is the number of labels extracted using that algorithm. In contrast, *recall* calculates the percentage of the human-assigned labels found by the text mining approach. It essentially measures how comprehensive the extracted labels are.

$$Recall = \frac{\sum_{i=1}^{C} M_i}{C}$$

Where C is the number of keywords in human assigned titles.

Using *precision* and *recall,* described above, *F-Score* is defined as,

$$FScore = \frac{2 * Precision * Recall}{(Precision + recall)}$$

In order to select the best labeling algorithm, we evaluate the performance of the algorithms in Section 3.2 in terms of *precision, recall, and F-Score.*

TABLE I. ACCURACY COMPARISON OF THE LABEL EXTRACTION ALGORITHMS

		Precision	Recall	F-Score
Min	*TFIDF*	0.08	0.14	0.11
	DeltaTF	0.17	0.21	0.17
	TFStDev	0.22	0.42	0.32
Avg	*TFIDF*	0.18	0.26	0.20
	DeltaTF	0.36	0.46	0.41
	TFStDev	**0.47**	**0.56**	**0.55**
Max	*TFIDF*	0.33	0.40	0.36
	DeltaTF	0.57	0.60	0.73
	TFStDev	0.67	1.00	0.75

Table 1 shows minimum, maximum and average values for *precision, recall* and *FScore* for each labeling algorithm. From the results, we can see that *DeltaTF* and *TFStDev* both produced better results than the *TFIDF* baseline. The *FScore* for *DeltaTF* was double that of the *TFIDF*, and *TFStDev* was almost triple. Of all three methods, *TFStDev* was by far the best.

To improve our understanding, consider the 6 level-3 children of the class "*Logic programming*". *TFIDF* produced *"xsb; program; thread"* as top three results out of which no terms match the actual titles. *DeltaTF* outputs *"logic; case; example"* as the top three labels for this cluster of which "logic" matches with the title but the remaining labels are unique. *TF-STDev* determined *"logic; program; constraints"* as top three results of which two match the original title terms, the maximum possible since the title only had two terms.

TABLE II. RESULTS OF DIFFERENT METHODS FOR LEVEL2 CLASS "PROGRAMMING TECHNIQUES"

CCS Class	TFIDF	DeltaTF	TFStDev
Automatic programming	gpf	**program**	partial
	tag	partial	**program**
	program	static	static
Applicative (functional) programming	procedure	type	**functional**
	reg	haskell	procedure
	stream	procedure	type
Object–oriented programming	system	class	class
	object	method	**object**
	language	**object**	**program**
Concurrent programming	actor	event	**concurrent**
	event	**concurrent**	event
	time	synchronization	synchroniza-tion
Logic programming	xsb	**logic**	**logic**
	program	case	**program**
	thread	example	constraints
Visual programming	user	user	user
	visualizer	menu	**program**
	cicsplex	interface	system

From the results we can see that, although all algorithms produce good results, the TFStDev labels overlap the true class names more often. We also observe that, the label *"program"* has appeared as high weighted term for four subclasses when using *TFStDev* even though that label is most commonly occurring term in all the classes. However, the word "*programming*" also occurs in many of the human-assigned class names as well.

IV. CONCLUSION

Stimulated by the need for a more comprehensive browsing interface for CiteSeer^X, we have developed automatic dynamic ontology expansion technique. In this work, we focus on a new approach for labeling the new subclasses and evaluated our approach with existing CiteSeer^X class labels. In particular, we compared the TFIDF, DeltaTF, and TFStDev methods on a set of 150 level-3 classes. Based on our results, we found that the TFStDev method performed the best. On average, it produced an average F-score of 55%. It suggested roughly half the words that were contained in human-assigned class labels and half of the labels it suggested were indeed in the class names.

Our future work will focus extending the work for to include noun phrases rather than single terms. We will also deploy our work on the CiteSeer^X site and evaluate the quality of our labels on subclasses created by our clustering techniques. These classes are likely to contain noisier collections of documents than those classes created manually so it will be important to see how well the methods work in practice.

V. ACKNOWLEDGEMENTS

This research is partially supported by the NSF grant number 0958123 - Collaborative Research: CIADDO-EN: Semantic CiteSeer^X.

REFERENCES

[1] P. Velardi, A. Cucchiarelli, and M. Petit, "A Taxonomy Learning Method and Its Application to Characterize a Scientific Web Community", IEEE Transactions on Knowledge and Data Engineering, 19 (2), Feb. 2007, pp. 180-191.

[2] M. Speretta, S. Gauch, "Automatic Ontology Identification for Reuse", International Conference on Web Intelligence, Nov. 2007, pp. 419-422.

[3] F. Geraci, M. Pellegrini, M. Maggini, "Cluster Generation and Cluster Labelling for Web Snippets: A Fast and Accurate Hierarchical Solution", Proceedings of the 13th international conference on String Processing and Information Retrieval, Glasgow, UK, 2006, pp. 25-36.

[4] Y. Nanhong, S. Gauch, Q. Wang, H. Luong, "An Adaptive Ontology based Hierarchical Browsing System for CiteSeerx", The Second International Conference on Knowledge and Systems Engineering, Hanoi, Vietnam, October 7-9, 2010, pp. 203-208.

[5] S. Gauch, D. Ravindran, A. Chandramouli, "KeyConcept: Conceptual Search and Pruning Exploiting Concept Relationships", Journal of Intelligent Systems, 19 (3), Sept. 2010, pp. 265-288.

[6] L. C. C. Carvalheria, E. Satoshi, "A method for semi-automatic creation of ontologies based on texts", Proceedings of the 2007 conference on Advances in conceptual modeling: foundations and applications, Auckland, New Zealand, 2007, pp. 150-159.

[7] D. Carmel, H. Roitman, and N. Zwerdling, "Enhancing cluster labeling using Wikipedia," Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, New York, USA, 2009, pp. 139-146.

[8] P. Treeratpituk, and J. Callan, "Automatically Labeling Hierarchical Clusters," Proceedings of the 2006 national conference on Digital government research, San Diego, USA, 2006, pp. 167–176.

[9] J. Evermann, J. Fang, "Evaluating ontologies: Towards a cognitive measure of quality," Information System, Elsevier Science Ltd., 35(4), 2010, pp. 391-403.

[10] ACM Classification System (CCS), 1998, [Online]. Available: http://www.acm.org/about/class/1998.

[11] G. Karypis, Cluto: Software for Clustering High-Dimensional Datasets, 2008, [Online]. Available: http://glaros.dtc.umn.edu /gkhome/cluto/cluto/download.

[12] M. Bhatt, A. Flahive, C. Wouters, W. Rahayu, D. Taniar, and T. Dillon, "A Distributed Approach to SubOntology Extraction", Proceedings of the 18th International Conference on Advanced Information Networking and Applications – Volume 2, IEEE Computer Society, Washington, DC, USA, 2004, pp. 636-.

[13] N. Niu, S. Reddivari, et al., "Automatic Labeling of Software Requirements Clusters", 4th International Workshop on Search-Driven Development, 2012, pp. 17-20.

[14] F. Fumiyo, Y. Suzuki, "Cluster Labeling Based on Concepts in a Machine-Readable Dictionary," Proceedings of 5th International Joint Conference on Natural Language Processing, Nov. 2011, pp. 1371-1375.

[15] A. Maedche, S. Staab, "Ontology learning for the Sematic Web", IEEE Intelligent Systems, IEEE, 16(2), 2001, pp. 72-79.

[16] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation", IEEE Trans. Pattern Analysis and Machine Intelligence, 24(7), 2002, pp. 881-892.

[17] P. Pantel, and D. Ravichandran, "Automatically Labeling Semantic Classes". In Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistics Conference, 2004, pp. 321-328.

[18] Y. Zhao, and G. Karypis, "Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering". Machine Learning, 55(3), 2004, pp. 311-331.

[19] D. Haverkamp, and S. Gauch, "Intelligent Information Agents: Review and Challenges for Distributed Information Sources", Journal of the Society for Information Science, 49(4), April 1998, pp. 304-311.

[20] A. Deitel, C. Faron, and R. Dieng, "Learning Ontologies from RDF annotations". In proceedings of IJCAI Workshop in ontology learning, Seattle, USA, 2001.

[21] OWL Web Ontology Language (2004), W3C Recommendation, 10 February 2004, [Online]. Available: http://www.w3.org/TR/owl-features/.

[22] Berners-Lee, Tim; James Hendler and Ora Lassila (May 17, 2001). "The Semantic Web". Scientific American Magazine. Retrieved March 26, 2008.