International Journal of Engineering Research And Management (IJERM)
ISSN : 2349- 2058, Volume-02, Issue-04, April 2015

Enhancing Text Mining by inducing Side-information with Ontology

Atiya Kazi , D.T.Kurian

Abstract— **There are several text mining applications where, the side-information contained within the text document may be considered as part of the clustering process. This side-information can be related to document provenance, which provides the author information, hyper links within the document to navigate between web-pages, user-access behavior pattern gathered from web logs, or images within the document. The proposed algorithm, which performs clustering of data along with the side information, combines classical partitioning algorithms with probabilistic models to increase the effectiveness of the clustering approach. The algorithm will be further extended to the classification problem to design a training model using the side-information. The proposed work will then generate target ontology, from saved user preferences and will then use the same in order to improve the effectiveness of text mining approaches. This will enhance the structuring of knowledge gained from text documents by incorporating the ontology as well as side-information. The advantage of incorporating side information over pure text-based clustering is that the side information can be vastly edifying, and it also contributes to creating more lucid clusters.**

Index Terms — Clustering, Text Mining, Ontology, Side-Information.

I. INTRODUCTION

Data Mining is the process of scrutinizing data from different perspectives and summarizing it to gain valuable knowledge. It comprises of clustering and classification on text based data, numeric data and web based data. In many application domains, a remarkable amount of side information is usually available along with the documents which is not considered during pure text based clustering[8]. Clustering text collections has been scrutinized under Data mining in [13]. Some efficient streaming techniques use clustering algorithms, that are adaptive to data streams, by introducing a forgetting factor that applies exponential decay to historical data [9]. Normally, text documents typically contain a large amount of meta information which may be helpful to enhance the clustering process. While such side-information can improve the quality of the clustering process, it is essential to make sure that the side-information is not noisy in nature. In some cases, it can hamper the eminence of the mining process. Therefore, one needs an approach which, carefully perceives the consistency of the clustering distinctiveness of the side information, along with the text content. The core approach is to determine a clustering process where text attributes along with the additional side-information provide comparable hints regarding the temperament of the basic clusters, as well as, they ignore conflicting aspects. The goal is to show that the reward of using side-information broadens the data mining process beyond a pure clustering task.

Recently, Ontologies have become an integral part of fabricating knowledge, so as to create knowledge-intensive systems. An ontology is formally defined as an explicit formal hypothesis of some domain of interest which helps in the interpretation of concepts and their associations for that particular domain [2]. To build any ontology, one needs a data mining expert who understands all the domain concepts, hierarchies and the relationships between them for a specialized domain. The current work focuses on techniques, which incorporate a user-preference ontology during the data mining process. It proposes a methodology for construction of an user-preference ontology, on the basis of the data stored in the mining log, generated after the classification of data. The effects of the generated ontology are studied for improving the data mining process.

II. RELATED WORK

The major work in the field of data mining looks upon scalable clustering of spatial data, data with boolean attributes, identifying clusters with non spherical shapes and clustering for large databases[7]. Several general clustering algorithms are discussed in [3]. An efficient clustering algorithm for large databases, known as CURE, has been covered in [14]. The scatter-gather technique, which uses clustering as its primitive operation by including liner time clustering is explained in [16]. Two techniques which develop the cost of distance calculations, and speed up clustering automatically affecting the quality of the resulting clusters are studied in [10]. An Expectation Maximization (EM) method, which has been around ages for, text clustering has been studied in [12]. It selects relevant words from the document, which can be a part of the clustering process in future. An iterative EM method helps in refining the clusters thus generated. In topic-modeling, and text-categorization, a method has been proposed in [11] which makes use, of a mathematical model for defining each category. Keyword extraction methods for text clustering are discussed in [10]. The data stream clustering problem for text and categorical data domains is discussed in [8]. Speeding up the clustering process can be achieved by, speeding up the distance calculations for document clustering routines as discussed in [15]. They also improve the quality of the resulting clusters. A similar approach is proposed in [5], which uses domain based, schema based, constraint based and user preference based ontologies for enhancing the test clustering process.

Manuscript received April 16, 2015
Atiya Kazi, Dept of Information Technology, RMDSSOE, Pune, India
D.T.Kurian, Dept of Computer Engineering, RMDSSOE, Pune, India

However, none of the above mentioned works with the combination of text-data with other auxiliary attributes. The previous work dealing with network-based linkage information is depicted in [6], [7], but it is not applicable to the general side information attributes. The current approach uses additional attributes from side information in conjunction with text clustering. This is especially useful, when the Side-information can regulate the creation of more consistent clusters. There are three forms of extending the process of knowledge discovery, with respect to their related ontologies, which are categorized as follows [4],

- Using on hand ontologies for knowledge discovery , during data mining.
- Construction of ontologies through knowledge discovery from mined results.
- Constructing and extending ontologies through knowledge discovery via existing ontologies.

The combination of the first two plays a major role in the methodology of the current paper.

III. The ontology building algorithm

The objective of an ontology based decision tree is to give a meaningful, descriptive and a readable view of concepts generated from associative rules, which will be stored in the mining log after the classification process. This ontology will be utilized by the users to retrieve the target data more efficiently. It will speed up the tuning process of the mining engine. Ontology is officially defined as follows, A core ontology is a structure consisting of two disjoint sets C and R whose elements are called concept identifiers and relation identifiers, respectively, a partial order called taxonomy $\leq c$, on C, a function $\sigma : R \rightarrow C^{+}$ called signature, a partial order $\leq R$ on R, called relation hierarchy, where $r1 \leq R\ r2$ implies $|\sigma(r1)| = |\sigma(r2)|$ and $\pi i(\sigma(r1)) \leq C\ \pi i(\sigma(r2))$, for each $1 \leq i \leq |\sigma(r1)|$ and C + is the set of tuples over C with at least one element and π_i is the i-th component of a given tuple. This definition can be expressed using (1).

$$O := (C, \leq c , R, \sigma, \leq R) \quad (1)$$

The algorithm initially chooses a set of decision nodes. For each node, there is a class created for every decision tree branch. Then a set of nodes is selected as tree branches by using a function Get-Branches. Then the hierarchy or the leaf nodes are generated by using the function GetLeaveBranch. These ontologies are created using OWL which stands for web based ontological language. The mapping between decision trees and ontologies can be expressed using the Figure 1. The algorithm makes use of a given decision tree as input to any web based ontological language to generate an user-preference ontology. The input consists of decision nodes, tree branches along with the target attribute set. The functions to get to all branches which include specific node and branches are also included as input. The final output will be, the generated user preference ontology, with filtering conditions based on the association rules which are extracted from the mining log. Each rule will have a corresponding support and confidence associated with it, to help a naïve user in the future for mining of the target data.

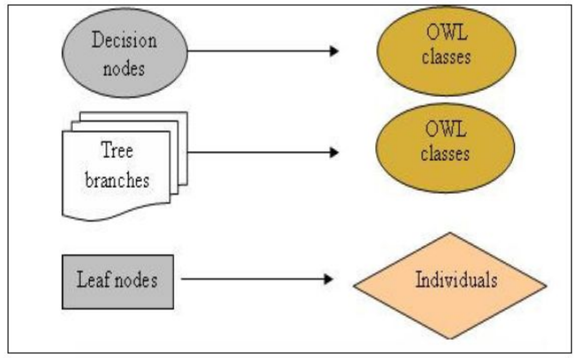

Figure 1. Mapping between decision trees and ontology

The algorithm for ontology building from a given decision tree can be expressed as follows,

Input:

- A decision tree.
- A set of decision nodes
- A set of distinct tree branches
- Target attributes set
- Get-Branches is a function to get to all branches which include specific node.
- GetLeaveBranch is a function to get to the branch of the leaf node.
- A function to get the class that represents every decision tree branch
- A function to create an individual object for every leaf node.

Output: ontology for a given decision tree

Method:

- Begin
- for each node N of decision-nodes
- Create a class using Class-labels
  - Add Attributes
  - Add Property to classes.
- for Each Branch B using Get-Branches,
  - Add Domain related to base class

  End for
- End for

//Steps to generate a class representing the target attributes

- Create Target Class
  - Add Attributes
  - Add Property to classes
- Generate Data-type property for target classes
  - Add domain to target attribute

//Steps to generate classes representing decision tree branches

- for each branch B of tree-branches
- Assign Class-labels
  - Assign class-Id
- End for
- For each node N
  - Assign node name
- End for

//Steps to represent the leaf nodes as individuals

- for each node of the decision tree
- Find leaves of every branch using GetLeaveBranch

- Create individual nodes for each leaf
- End for
- End

IV. Data Mining using Ontologies

The inception of ontologies and the data mining process portray a cyclic behavior pattern[4]. There can be a broad set of interactions between the naïve user, the mining engine and the user preference ontology. The knowledge engineer may start with primitive methods which require very little or even no background knowledge, to return only simple values, like term frequencies. While the knowledge mining model matures during the course of time, the user may turn towards more advanced and more knowledge-intensive algorithms, such as the ones described in the current work. The data mining expert uses the document preprocessing component to initiate the knowledge discovery process by choosing among a set of text pre-processing methods available in the system. The output will be the target data sent to the mining engine. The mining engine will employ term extraction mechanisms to extract relevant terms from the corpus. The Ontology modeling environment supports the data mining expert in adding newly discovered conceptual structures to the ontology with the help of the user-preferences stored in the mining log. In addition to core capabilities for structuring the ontology, the environment also provides some additional features for the purpose of documentation, maintenance, and ontology exchange. The system will then internally store the ontologies using an XML serialization of the ontology model. The Figure.2 highlights the principle idea of a cyclic framework which is based on knowledge discovery methods. The objective of this research work is to utilize ontology as background knowledge, to document clustering process, and observe the effects on the clustering performance of several datasets. It seizes the output generated from mined data to generate an ontology, based on user-preferences.

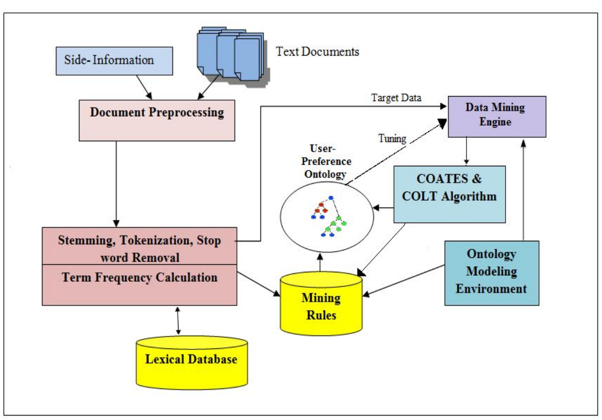

Figure 2. The cyclic process of ontology based data mining with side-information

This work aims at proving that, ontology based clustering algorithm is better than the traditional clustering schemes. The ontology thus generated, will help the user in performing mining in a more efficient way. The system intends to help a naïve user with extracting relevant information by building user-preference ontology. The creation of knowledge is encouraged by implicit settings of the data models, and the enrichment of data is fashioned by conceptualization of explicit user preference ontologies. The algorithms used for performing data mining tasks such as clustering and classification, by including side-information, are COATES (Content and Auxiliary attribute based Text clustering algorithm) and COLT (Content and Auxiliary attribute based Text classification algorithm) respectively[1].

The COATES algorithm is initialized using, *k* number of clusters. Stop-words should be removed beforehand, and stemming should be performed in order to improve the biased power of the attributes. The Term frequency should also be calculated in this stage to prune out the irrelevant words from the documents. The two major phases of the above algorithm are explained below:

• **Initialization:** This is a text only phase, and does not include any side information. The centroids and the partitioning based on them, created by the clusters formed in this phase, provide the trigger for the next phase of clustering.

• **Main Phase:** This phase starts off with the initial clusters of the previous phase, and performs two minor iterations which include, the text content as well auxiliary attribute information, so as to improve the clustering quality.

The COLT algorithm relies on a supervised clustering approach, to segregate the data into *k* distinct clusters. This segregation of data, handles the extended classification process. The training algorithm follows the following steps :

• **Feature Selection**: Feature selection is necessary to remove both, text and the auxiliary attributes, which are not associated with the class label.

•**Initialization:** This step employs a modified k-means approach to initialize clusters, using purely text content, so that each cluster has the records of a particular class only.

•**Cluster-Training Model Construction:** This phase combines both text and side-information, for creating a cluster-based model. The set of supervised clusters are used for the classification process. A decision tree can be generated using the training model based on class labels associated with each cluster.

Pruning out the noisy side-information

The noisy attributes of the side-information, need to be eliminated, to enhance the robustness of the approach. This is achieved by computing the gini-index value for an attribute r. This is done at the beginning of each auxiliary iteration, based on the clusters from the previous content-based iteration. The gini-index of an attribute r, for a given set of k clusters, is calculated using (2):

$$G_r = \sum_{j=1}^{k} P_{rj}^{2} \qquad (2)$$

Where, P_{rj} is the relative presence of attribute r in cluster j denoted by (3),

$$P_{rj} = f_{rj} \;/\; \sum_{j=1}^{k} f_{rm} \qquad (3)$$

Here, f_{rj} is the fraction of the records in cluster C_j for which attribute r takes a value of 1. The value of G_r varies from 1 / k to 1. Consider only those attributes whose gini-index is above a threshold denoted by γ . The value of gini-index will vary from iteration to iteration based on the auxiliary attributes of that iteration.

V. Experimental Evaluation

The main aim is to show how the proposed approach provides superior clustering techniques by incorporating side-information along with text documents. The experimental evaluation of the proposed method can be carried out by computing the cluster purity of each generated cluster. Then, the average cluster purity will judge the quality of clustering process. The formula for calculating cluster purity(CP) can be denoted using (4),

$$CP = \sum_{i=1}^{k} C_i \ / \ \sum_{i=1}^{k} n_i \quad (4)$$

where, k denotes the generated clusters, n_i denotes data points in the clusters and C_i denotes the number of data points with a majority input cluster label Li. The cluster purity will lie in the range of 0 to 1, clearly indicating perfect clustering if it is 1 and poor clustering if it is 0.

Conclusion

The algorithms discussed in the current work propose a novel technique to enhance text clustering in conjunction with side-information. The benefit of using such an approach over traditional pure text-based clustering is, the side information is highly edifying, and contributes to creating more lucid clusters. The new approach also extends to include, an user-preference ontological schema, during the mining process for improving the clustering efficiency. This will help the mining engineer to visualize the conceptual knowledge as ontologies, which are easier to interpret and integrate. These ontologies can be used to prompt the surfacing of auxiliary knowledge from the data.

Acknowledgment

First and foremost, I would like to express my sincere gratitude to my guide, Ms. D. T. Kurian, for her unremitting support. I will always be grateful for her patience, motivation, enthusiasm, and immense knowledge. This work is an extension of the previous research efforts of Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. I am very much indebted to them for their inspiring work which is a boost for future researchers. Finally, I would like to pay my respect and love to my parents, for their encouragement throughout my career. .

References

[1] C. C. Aggarwal et al, “On the use of side-information for mining text data”, IEEE Trans. Knowl. Data Eng, vol 26, pp. 1415-1429, June 2014
[2] Henrihs Gorskis, Yuri Chizhov, “Ontology Building Using Data Mining Techniques”, Information technology and management science, vol 15, pp 183-188, 2013.
[3] C. C. Aggarwal and C.-X. Zhai, “A survey of text classification algorithms," in Mining Text Data. New York, NY, USA: Springer, 2012.
[4] Mathieu d’Aquina, Gabriel Kronbergerb, and Mari Carmen Suárez-Figueroa, “Combining Data Mining and Ontology Engineering to enrich Ontologies and Linked Data”, Proc. first International workshop on knowledge discovery and Data Mining , pp 19-24, 2012.
[5] Chin-Ang Wu et al., “Toward Intelligent Data Warehouse Mining: An Ontology-Integrated Approach for Multi-Dimensional Association Mining”, Information Technology and Management Science, Expert Systems with applications, volume 38, Issue 9, pp 11011-11023, sept-2011.
[6] J. Chang and D. Blei, “Relational topic models for document networks”, in Proc. AISTASIS, Clearwater, FL, USA, 2009, pp. 81-88.
[7] R. Angelova and S. Siersdorfer, “A neighborhood-based approach for clustering of linked document collections”, in Proc. CIKM Conf., New York, NY, USA, 2006, pp. 778–779.
[8] C. C. Aggarwal and P. S. Yu, “A framework for clustering massive text and categorical data streams”, in Proc. SIAM Conf. Data Mining, 2006, pp. 477-481.
[9] S. Zhong, “Efficient streaming text clustering”, Neural Netw., vol. 18, no. 5–6, pp. 790–798, 2005.
[10] Y. Zhao and G. Karypis, “Topic-driven clustering for document datasets”, in Proc. SIAM Conf. Data Mining, 2005, pp. 358-369.
[11] C. C. Aggarwal, S. C. Gates, and P. S. Yu, “On using partial supervision for text categorization,” IEEE Trans. Knowl. Data Eng.,vol. 16, no. 2, pp. 245–255, Feb. 2004.
[12] T. Liu, S. Liu, Z. Chen, and W.-Y. Ma, “An evaluation of feature selection for text clustering”, in Proc. ICML Conf., Washington, DC, USA, 2003, pp. 488-495.
[13] M. Steinbach, G. Karypis, and V. Kumar, “A comparison of document clustering techniques,” in Proc. Text Mining Workshop KDD, 2000, pp. 109–110.
[14] S. Guha, R. Rastogi, and K. Shim, “CURE: An efficient clustering algorithm for large databases”, in Proc. ACM SIGMOD Conf., New York, NY, USA, 1998, pp. 73-84.
[15] H. Schutze and C. Silverstein, “Projections for efficient document clustering”, in Proc. ACM SIGIR Conf., New York, NY, USA, 1997, pp. 74-81.
[16] D. Cutting, D. Karger, J. Pedersen, and J. Tukey, “Scatter/Gather: A cluster-based approach to browsing large document collections”, in Proc. ACM SIGIR Conf., New York, NY, USA, 1992, pp. 318-329.