Interpretable and Reconfigurable Clustering of Document Datasets by Deriving Word-based Rules

Vipin Balachandran
Indian Institute of Technology
Madras, Chennai, INDIA
vipin.bl@gmail.com

Deepak P
IBM Research - India,
Bangalore, INDIA
deepak.s.p@in.ibm.com

Deepak Khemani
Indian Institute of Technology
Madras, Chennai, INDIA
khemani@iitm.ac.in

ABSTRACT

Clusters of text documents output by clustering algorithms are often hard to interpret. We describe motivating real-world scenarios that necessitate reconfigurability and high interpretability of clusters and outline the problem of generating clusterings with interpretable and reconfigurable cluster models. We develop a clustering algorithm toward the outlined goal of building interpretable and reconfigurable cluster models; it works by generating rules with disjunctions and conditions on the frequencies of words, to decide on the membership of a document to a cluster. Each cluster is comprised of precisely the set of documents that satisfy the corresponding rule. We show that our approach outperforms the unsupervised decision tree approach by huge margins. We show that the purity and f-measure losses to achieve interpretability are as little as 5% and 3% respectively using our approach.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Clustering*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text Analysis*; I.5.4 [**Pattern Recognition**]: Applications—*Text Processing*

General Terms

Algorithms, Experimentation

1. INTRODUCTION

Text Clustering is the process of grouping text documents into clusters so that the documents within a cluster are more similar than documents across clusters (in the absence of label information). Similarities between text documents are often assessed using the cosine similarities between TF-IDF[1] vectors. Popular techniques for text clustering include partitional clustering algorithms such as K-Means [11] and hierarchical clustering algorithms such as Single-Linkage Clustering among others [17]. It is often necessary to interpret the clusters generated, for knowledge discovery in various scenarios. However, traditional clustering algorithms, when applied to text data, output models which are hard to interpret. For example, K-Means clusters may be represented as a set of cluster centroids where each document could be associated with the cluster whose centroid is most similar to it, according to the chosen similarity measure. The Vector Space Model for representing text data, however, has as many dimensions as the vocabulary of the collection; this makes cluster centroids hard to understand and interpret. The centroid vector is often criticized to be not very meaningful to describe clusters [4]. Popular text clustering toolkits such as CLUTO[2] aid visualization by tagging each cluster with descriptive and discriminative features. On the other hand, algorithms for co-clustering, those which simultaneously cluster objects and attributes (i.e., documents and words, in the case of text) [5] have been adapted to text data to generate sets of words along with every cluster of documents [12]. However, a set of words associated with each cluster is not a self-contained model as it does not fully describe a cluster. For example, given a set of clusters and sets of words for each, a document's membership in a specific cluster is not obvious (since a document may contain words from across two cluster descriptions, and may belong to a third). Further, such a model is not reconfigurable, i.e., the human cannot edit the model so that the clustering for the reconfigured model can be computed easily, or is obvious.

In this paper, we deal with the problem of clustering document datasets to derive **highly interpretable** cluster models which are **self-contained** and **reconfigurable**. We provide an approach for Rule Generating Clustering (RGC) towards this goal and show that the accuracy loss to achieve interpretability is very small.

2. RELATED WORK

While there is no strict criterion to decide on whether a clustering is interpretable or not, rule-generating models are generally considered to be interpretable [14]. Interpretability (e.g., using Mamdani-style rules) is the unique selling point of fuzzy systems [13] and is often achieved by trading off for some accuracy [8]. On the same lines as [13], the length-accuracy trade-off for cluster descriptions has been addressed previously (e.g., SOR model [6]). The high dimensionality of the document space renders SOR inappropriate.

Text clusters are often presented to the users as a word-cloud, sets of representative words, a concept taxonomy, word clusters derived during clustering (e.g., [5]) or summary words. We refer to such easily interpretable models as *word-based representations (WBRs)*. WBRs, however (as mentioned earlier), do not fully describe the cluster and are hence not self-contained/comprehensive. For example, in a text cluster that mostly contains sports news re-

[1] http://en.wikipedia.org/wiki/Tf-idf

[2] http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

CIKM'09, November 2–6, 2009, Hong Kong, China.

ports along with a few reports about celebrities who attended a specific sports event, none of the above may generate a word in the description that is indicative of the latter. Thus, an analysis of the WBR corresponding to that cluster could lead one to conclude that the cluster has only sports reports. Further, all documents in the cluster are not guaranteed to contain any one of the words included in the representation. WBRs are mostly read-only models; i.e., editing of such models do not lead to an intuitive reconfiguration of cluster memberships. A simplistic model of deleting documents containing the word when the word is deleted from the cluster model would not work since documents that do not contain any words in the description are also part of the cluster. WBRs are hence not self-contained nor do they allow easy manual reconfiguration.

2.1 Rule-Based Clustering Models

A standard approach to summarize multi-dimensional points is to represent them by a set of hyper-rectangles [1] e.g., $(3.80 \leq GPA \leq 4.33 \wedge 0 \leq minutes\_in\_gym\_per\_week \leq 30)$ may describe a set of *nerds*. *Sum of Rectangles (SOR)*, the canonical format for cluster descriptions in databases has been used for numerical data [6]. Text data is unique in being highly multi-dimensional and extremely sparse. SOR representation works by discovery of bounds and convex structures; the inherent high dimensionality and sparsity of document datasets makes upper bound discovery impractical. Decision trees have been adapted for clustering [2] to generate rules based on attribute frequencies.

2.2 Applications

Interpretability in machine learning models has been studied in various contexts [13, 6, 2] and its need cannot be overemphasized. Now, we describe a scenario where reconfiguration of cluster memberships is highly desirable, and elaborate on other real-world scenarios that demand or are benefitted by interpretability and reconfigurability of cluster models.

Service Delivery Organizations (SDO): SDOs mostly operate by providing support to solve customer issues, and are bounded by SLAs[3]. Resolution of each issue is guided by manually authored documented procedures (e.g., call flow charts). Each such issue is recorded in the form of a problem/change ticket whose contents are mostly textual. In current practice, managers use text clustering tools to cluster tickets that resulted in SLA violations (the most important quality indicator), and analyze such clusters using word-based representations such as tag clouds to identify distinct categories of problems that led to SLA violations. Seemingly problematic clusters are then given to *Quality Analysts* who analyse the clusters by reading and assimilating the tickets in those and provide insights to enable faster resolutions of such problems. Here, we would want to minimize the number of irrelevant documents in such clusters since they would contribute only marginally (or not at all) to derive insights from the cluster. Rule based interpretable models, being self contained, could boost the chances of being able to filter out such cases by just glancing at the rules. *In such a setting, usage of word-based representations is counter-intuitive since we want to remove statistically insignificant concepts, that are precisely the ones least likely to be represented in WBRs*.

Other Applications: The problem outlined above is a manifestation of a more general class of scenarios where users inspect large document datasets and select a few clusters for closer manual inspection. This poses the challenge of being able to refine the clusterings at any cost, since the laborious process of inspecting documents manually is the target of optimization. Other contexts include compiling a targeted news report by selectively reading certain categories of newswire reports, selecting customers to send targeted ads for a product, and recommending films to targeted groups using a collection of film reviews, each of which require careful manual post-processing of selected clusters to derive insights.

[3]http://en.wikipedia.org/wiki/Service_level_agreement

3. RULE-GENERATING CLUSTERING (RGC)

We propose an approach for Rule-Generating Clustering that partitions the dataset into *non-overlapping clusters of documents where each cluster has an associated rule that is satisfied only by the documents in that cluster*. For a document dataset D comprising of documents $\{d_1, d_2, \ldots, d_n\}$ with an associated vocabulary $W = \{w_1, w_2, \ldots, w_m\}$, RGC would yield k clusters $\{C_1, C_2, \ldots, C_k\}$ with each cluster C_i having n_i documents would have a rule R_i associated with it. Each rule R_i is a composite condition composed of atomic conditions that relate to the frequency of individual words. An example could be:

$$R_i = (f(w_{i1}) \geq 1) \vee (f(w_{i2}) < 1) \vee \ldots$$

This signifies that a document that contains at least one occurence of w_{i1} or does not contain w_{i2} would be part of C_i. The key difference from some well-known clustering algorithms is that some documents C_D in D may still be unclustered since they do not satisfy any cluster's rule.

Centroid similarity Ranking (CR):A novel feature selection method forms the skeleton of the RGC approach. Traditional feature selection methods such as tf-idf, term contribution [10] rank features such that the top few features would be able to cover most of the documents in the dataset and that a clustering using such top features would lead to high accuracy when compared against available extrinsic labels. This is different from our objective of selecting highly *pure* words; words such that most of the documents containing them are homogeneous. Every attribute w has an associated set of documents D_w, the set of documents containing w. Documents are expressed as tf-idf vectors and let the centroid vector of D_w be denoted by $\bar{cv}_w$. The homogeneity value of a word w is the fraction of documents in D_w that are closer to $\bar{cv}_w$ than any other centroid $\bar{cv}_v$ for any word v that has presence in at least 1% of the documents. Formally,

$$|\{d \in D_w : \forall_{v \neq w}(Sim(d, \bar{cv}_w) \geq Sim(d, \bar{cv}_v))\}|/|D_w|$$

where $Sim(.,.)$ denotes the cosine similarity of the argument vectors. CR prioritizes words with high homogeneity values. Like any other clustering approach, other similarity measures could be employed to form variants of the approach.

The different phases of the algorithm are:

1. **Cluster Generation:** This phase uses CR to select top-t words and builds one cluster per word using the documents that contain that word. All rules at this phase are of the form $f(w) \geq 1$.

2. **Merging of Clusters:** This phase merges the most similar pair of clusters (in agglomerative fashion [9]) to generate a single bigger rule associated with the merged cluster, until there are exactly k clusters. The clusters chosen for merge may have overlaps. This is dealt with, by adding negated conditions as illustrated in Example 1.

3. **Coverage Enhancement:** This attempts to bring in more documents into the clustering, thus enhancing coverage. This is done by choosing from among the set of words not yet included in the clustering; if the set of documents containing

Alg. 1 RGC-N

1: $C \leftarrow \phi$, $R \leftarrow \phi$,W_α = *top-t words acc. to CR*
2: **for** $w \in W_\alpha$ **do**
3: $C = [C, \{w\}], R = [R, \{f_w \geq 1\}]$
4: **end for**
5: **while** $|C| < k$ **do**
6: *Remove the most similar pair of clusters* c_i, c_j *from* C
7: *Avoid_Overlap(*c_i, c_j*)*
8: *Merge them and add them to* C
9: **end while**
10: *Reduce_Cluster_Rules(C,R)*
11: **while** W *has words yet to be considered* **do**
12: *pick* $w \in W$ *where* $|D_w - C|$ *is maximum*
13: **if** D_w *overlaps with the cluster with which it has maximum similarity* **then**
14: *Merge* D_w *with that cluster*
15: **end if**
16: **end while**

the chosen word overlaps with only one cluster, and if it bears maximum similarity with that cluster, a merger is performed.

The Algorithm (Algorithm 1) identifying W_α, the top-t words according to CR that covers at least $\alpha\%$ of the dataset. The cluster generation phase (lines 2-4) generates one cluster out of each word in W_α, the cluster comprising of all words containing at least one occurence of the word. It may be noted that these clusters need not be disjoint and may overlap. The Merging phase (lines 5-10) starts off with multiple overlapping clusters, and merges them to k clusters, eliminating overlaps when necessary (as described in Example 1). The Merging phase may introduce some redundancy in the rules due to merging to a cluster to which overlap was avoided earlier; here, the negated condition added earlier could be eliminated due to the merger (as illustrated in Example 1). The $ReduceClusterRules(.)$ function eliminates such redundancies in an easy and straightforward way. The coverage enhancement phase (lines 11-15) considers remaining words, picking those words that occur in most unclustered documents first. Each such word that has a single overlapping cluster would have itself added to the cluster if it bears maximal similarity with that cluster. Each such merger leads to a merger of the corresponding rules using a disjunction.

Example 1: Consider two clusters c_1 and c_2 chosen for merger, represented by rules $f_{w_1} \geq 1$ and $f_{w_2} \geq 1$ respectively. Let it be the case that c_1 overlaps with another cluster c having a rule $f_w \geq 1$. Since we want to eventually have non-overlapping clusters in the output, we resort to avoiding the overlap between the merged cluster and c. In RGC-N, we accomplish this merger by forming a new cluster merging c_1 and c_2 but excluding those documents that are in c. This leads to the following rule:

$$(f_{w_1} \geq 1 \wedge !(f_w \geq 1)) \vee f_{w_2} \geq 1$$

The condition $(f_{w_1} \geq 1 \wedge !(f_w \geq 1))$ represents $c_1 \backslash c$, which is then merged (using disjunction) with c_2. This negated condition may be removed later if the merged cluster and c become part of a single cluster by mergers later on. ■

4. OTHER APPROACHES

UDT [2] is an approach for interpretable clustering that generates text clusters that could be represented by rules on word frequencies, and hence, is fully reconfigurable. This decision tree based technique works by starting with the entire corpus as the dataset associated with the root node, and progressively splits it into child nodes using word frequency conditions in a recursive operation. The information gain guided splitting continues as long as node sizes are larger than a threshold. This hierarchical clustering approach differs from RGC since it could generate much more than k leaf clusters. It may be specifically noted that UDT's only parameter, the threshold on the maximum size of the dataset at a leaf node is very different from the k parameter for RGC and popular clustering algorithms such as K-Means.

5. EXPERIMENTAL EVALUATION

We now empirically analyze the proposed approaches against UDT and C-RG. We first describe the datasets and the evaluation metrics that we use in our experiments. In the subsections that follow, we describe the results for the various analyses performed and sample rules generated from the vatrious techniques.

5.1 Datasets and Evaluation Measures

The datasets that we use for our evaluation are listed in Table 1. These datasets were previously used for validating document clustering algorithms in [16]. While $Sports$ and $K1b$ are entire datasets, the rest are subsets of datasets described in [16]; these subsets were chosen to ensure a wide variety in total dataset sizes, as well as in the average number of documents per class. We use the default setting of CLUTO toolkit for the K-Means experiments.

We evaluate the quality of the various clustering algorithms against extrinsic document labels that are available with the datasets; each document is assigned to a unique specific class in each of the datasets that we have considered. Further, we also evaluate the interpretability of the rule bases generated by the algorithms using the average length of the rules generated. The extrinsic document labels available are referred to as class labels hereon. We employ the following measures in our evaluation:

1. **(Net) Purity:** The purity of a cluster is defined as the fraction of the maximally represented class in a cluster. The net purity is computed as the weighted average of purities across clusters, weighted by their cardinalities. For RGC, unclustered documents are assumed to be misclustered.

2. **F-Measure:** F-Measure [15, 3, 7] combines the precision and recall ideas from information retrieval. For clustering evaluation, each class is treated as a query, and each cluster is treated as the result of the query. The f-measure for the class is then the harmonic mean of the precision and recall.

3. **Rule Length:** Each rule generated by interpretable clustering algorithms that we consider are composed of atomic conditions that express a frequency threshold based condition. We refer to the length of a rule as the number of atomic conditions that it has.

5.2 Comparison with UDT

UDT typically generates much more leaf clusters than k. It is desirable that the number of clusters in the output be close to the number of classes (i.e., k); else purity values may not be meaningful. For example, a clustering that puts each document in its own cluster would have a net purity of 1.0. This makes an absolute comparison between our algorithms (that always generate only k clusters) and UDT inappropriate. On the same lines, the average rule length is also not meaningful to consider for comparison since the denominator for averaging is much higher for UDT as compared to our algorithms. *However, the total rule length across clusters is still indicative of the interpretability of the clustering.* Larger rules are intuitively harder to interpret. Figure 1 shows that

Table 1: Datasets Used

Dataset	Documents	Words	#Classes
Sports	8580	18324	7
K1b	2340	21839	6
Ohscal3	2864	11465	3
R4	1013	7015	4
C3	3893	15490	3
Cranmed	2431	41681	2

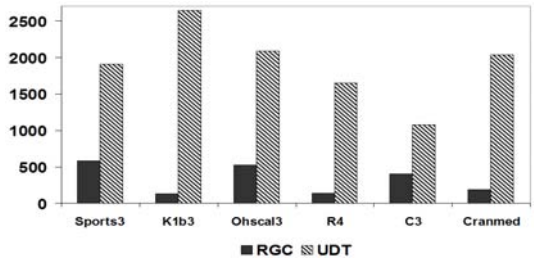

Figure 1: Total Rule Lengths.

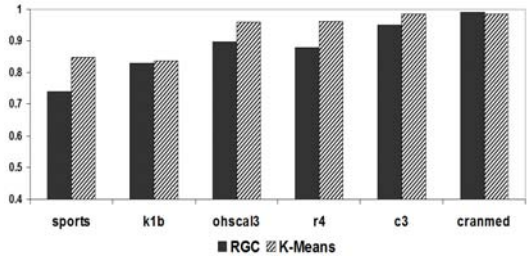

Figure 2: Net Purity.

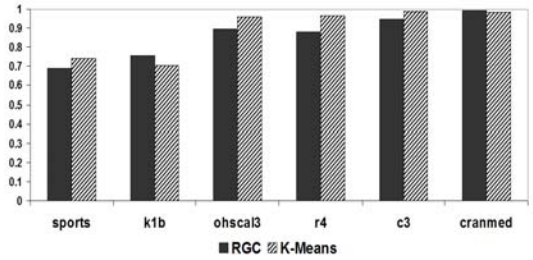

Figure 3: F-Measure.

RGC outperforms UDT by almost 4 times; the total rule lengths of RGC and UDT stand at 494 and 1946 respectively. Although UDT seemingly gives high purity, such high purities are achieved with as many as 60-90 clusters and are hence not very useful. Evaluations based on other measures such as Entropy and F-Measure also were found to assert that RGC and RGC-N are much better than UDT. In summary, RGC is seen to empirically perform much better than UDT for text clustering.

5.3 Comparison with K-Means

Now, we analyze RGC and its performance against the K-Means algorithm. High values of net purity and f-measure are desirable whereas lower values of entropy indicate that the clustering corresponds better to the labels. The Purity of the clusterings generated are illustrated in Figure 2. K-Means performs consistently better than RGC leading to an average purity of 0.93 wheras the RGC clusterings have a purity of 0.88 on the average. The F-Measure plot (Figure 3) also reveals a similar behavior with RGC faring 0.03 lesser than K-Means clusterings on an average.

6. SUMMARY AND CONCLUSIONS

Document clustering techniques are well evolved and give very high accuracies, but often produce models that are hard to interpret. However, in certain real world scenarios where clusters of documents are to be selected for manual review, it becomes necessary to produce an interpretable and reconfigurable model of the clustering. A variety of such scenarios exist. Most document clustering algorithms in literature score poorly on the combined goal of interpretability and reconfigurability of cluster models. Our approach for interpretable document clustering, RGC, associates each cluster with a rule of conditions on word frequencies; the rule is satisfied by only those documents that belong to the cluster. However, such an approach could leave out some documents as unclustered. An empirical evaluation against UDT illustrates the effectiveness of our approaches. Our analysis of the well studied accuracy-interpretability trade-off in the context of RGC shows that the RGC clusterings are only at most 5% less pure than those from classical clustering algorithms for a wide variety of text datasets.

7. REFERENCES

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD*, 1998.

[2] J. Basak and R. Krishnapuram. Interpretable hierarchical clustering by constructing an unsupervised decision tree. *IEEE TKDE*, 2005.

[3] F. Beil, M. Ester, and X. Xu. Frequent term-based text clustering. In *KDD*, pages 436–442. ACM, 2002.

[4] D. Boley. Hierarchical taxonomies using divisive partitioning. Technical report, 1998.

[5] I. Dhillon, S. Mallela, and D. Modha. Information-theoretic co-clustering. In *KDD*, 2003.

[6] B. Gao and Ester. Cluster description formats, problems and algorithms. In *SIAM Intl. Conference on Data Mining*, 2006.

[7] A. Hotho, S. Staab, and G. Stumme. Ontologies improve text document clustering. In *ICDM*, pages 541–544, 2003.

[8] H. Ishibuchi and Y. Nojima. Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning. *Int. J. Approx. Reasoning*, 2007.

[9] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 1999.

[10] T. Liu, S. Liu, Z. Chen, and W. Ma. An evaluation on feature selection for text clustering. In *ICML*, 2003.

[11] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *5th Sym. of Maths, Statistics and Probability*, 1967.

[12] B. Mandhani, S. Joshi, and K. Kummamuru. A matrix density based algorithm to hierarchically co-cluster documents and words. In *World Wide Web Conference*, 2003.

[13] D. D. Nauck. Measuring interpretability in rule-based classification systems. In *ICFS*, 2002.

[14] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[15] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques, 2000.

[16] Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis. In *TR, Univ. of Minnesota*, 2001.

[17] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *DMKD*, 2002.