Unsupervised Methods for Resolving Mixed Entities on the Web

Ingyu Lee
Sorrell College of Business
Troy University
Troy, AL. USA

Hongrae Lee
Department of Computer Science
University of British Columbia
Vancouver, BC. Canada

Byung-Won On
School of Information Systems
Singapore Management University
Singapore

***Abstract*—In the World Wide Web, Mixed Entity Resolution is a common problem due to their homonym when non-unique values are used as the identifier. For example, if only surname is used as an identifier, one cannot distinguish "Vanessa Bush" from "George Bush." To resolve such mixed entities on the Web, we develop unsupervised methods that approximate the number of clusters (known as *numClus*), and then combine various supervised clustering algorithms including K-means, Spectral Bisection, Latent Dirichlet Allocation, and Similarity Propagation. Experimental results show that in addition to the improvement of about 16% accuracy of clustering results at best case, *numClus* identified the number of clusters ten times more correctly than Hierarchical Clustering.**

***Keywords*-data mining; clustering; mixed entity resolution**

I. Introduction

Recently U.S. Census Bureau reports that about 30% queries include person names and 100 million persons share only about 90,000 person names. The report supports our claim that a search result is a mixture of web pages of individuals with namesakes. For instance, [1] showed a search result containing a mixture of web pages of a professor at CMU, an actor, a hockey player, a historian, a Jazz guitarist, etc., who have the same name spellings of "Tom Mitchell." Actually, there exist 37 different *Tom Mitchell*s among 100 top ranked web pages retrieved by Google search engine. This is called a mixed entity resolution problem and is defined as the followings:

> Given a set of mixed entities E=$\{e_1, ..., e_p, ..., e_q, ..., e_N\}$ with the same name description d, group E into K disjoint clusters C=$\{c_1, ..., c_K\}$ such that entities $\{e_p, ..., e_q\}$ within each cluster c_i belongs to the same real-world group.

Clustering is the key part for this task. In particular, we view this problem as an **unsupervised clustering problem** which can be addressed by either partitioning or hierarchical clustering. Hierarchical agglomerative clustering approaches generate a series of nested clusters by merging simple clusters into larger ones, while partitive methods try to find a pre-specified number of clusters that best capture the data. For instance, a prior knowledge of the probable number of clusters must be required in K-means and K-way spectral clustering algorithms. In our problem, *Hierarchical Agglomerative Clustering* (HAC) algorithms are mainly considered as a solution because the correct number of clusters is not given *a priori*. However, since hierarchical clustering methods are not able to reallocate entities, it is plausible to be poorly classified in the early stages of text analysis [9], [14]. In this paper, we present an approximated algorithm to measure the number of clusters, combined with various supervised clustering methods that outperform hierarchical clustering methods.

Algorithm 1: numClus: estimating number of clusters.

Input : Graph G
Output: # of clusters

for $\mu = 0.01; \mu \leq 1; \mu = \mu + 0.01$ **do**
  Generate a subgraph G_i by removing;
  links s.t. their link weights $\leq \mu$;
  $numClus_i$ = # of disconnected graph segments in G_i;
  L=(μ,$numClus_i$);
Based on the L sequences, classify as Type I (concave), Type II (convex), or Type III (linear);
Find the maximum and minimum difference between $numClus_i$ and $numClus_{i+1}$ in L;
if *graph segments* $\{G_i\}$ *are in Type I* **then**
  Return maximum difference $numClus_i$ as cluster #;
else if *graph segments* $\{G_i\}$ *are in Type II* **then**
  Return minimum difference $numClus_{i+1}$ as cluster #;
else if *graph segments* $\{G_i\}$ *are in Type III* **then**
  Return average of maximum and minimum difference;

This paper consists of the followings. Section II describes about the main proposed algorithm including the algorithm to estimate the number of clusters. Experimental results with public name data set are followed in Section III. Related works are described in Section IV. Section V includes concluding remarks and future plans.

II. Main Proposal

Our unsupervised method consists of two steps. In the first step, the number of clusters is approximately estimated by Algorithm 1. Then, four supervised clustering methods are used with the approximated number of clusters. In addition, note that we comparatively study various supervised clustering methods – (1) K-means based on distance measurement, (2) Spectral Bisection using eigenvector properties, (3) Sim-

978-981-08-6846-8/$26.00

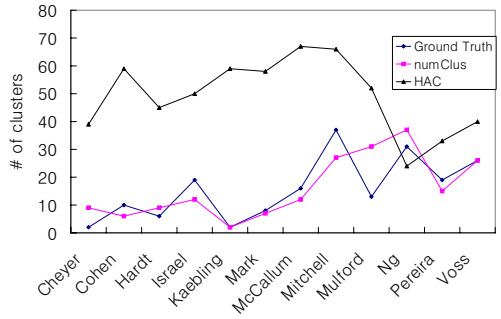

Figure 1. Results of approximating # of clusters in 12 name data sets [1].

ilarity Propagation based on responsibility and availability between nodes, and (4) Latent Dirichlet Allocation based on joint distribution of topics.

A. *numClus: Estimation of the Number of Clusters.*

To estimate the number of clusters, a number of approaches have been proposed. In particular, recent, leading studies are Gap Statistic [11] and Clest [10]. However, such methods have been developed to address the problem in which the cluster sizes are well distributed. In addition, they focus on estimating a small number of clusters (e.g., at most 2 or 3 clusters). Those methods are not efficient for mixed entity resolution which has extremely skewed sizes of clusters and a large number of clusters [4], [8]. To overcome the latter, we proposed an algorithm, named *numClus*, to estimate the number of clusters using similarity values between Web pages.

Initially, web pages are represented as a graph G, where the p-th web page denotes a node e_p, and the edge weight between nodes e_p and e_q is the TF/IDF cosine similarity between e_p and e_q. Then, we propose an algorithm of approximately estimating the number of clusters illustrated in Algorithm 1. The main idea of the algorithm is to gradually disconnect several edges based on the connectivity between nodes and to analyze the patterns of segmented subgraph sequences. Assume N_{μ_i} and $N_{\mu_{i+1}}$ are number of subgraphs with μ_i and μ_{i+1}, respectively. Then, we assume that the number of clusters are approximately located when the difference between N_{μ_i} and $N_{\mu_{i+1}}$ is drastically changed. We observe three different types of graph segment sequences with an incremental threshold value μ: gradually increasing, gradually decreasing, and staying approximately as a constant. In the gradually increasing sequence (convex shape), the sequence reaches the maximum difference when the difference starts to decrease. For the gradually decreasing sequence (concave shape), the sequence reaches the minimum difference when the difference starts to increase. Based on this property, our algorithm assumes that the number of cluster will be close to the maximum difference in the number of subgraph sequences when it shows concave function, the minimum difference when it shows convex function, and the average of these two when it stays approximately as a constant (linear function).

B. *Four Supervised Clustering Methods.*

K-means repeatedly computes the distance from centroids and assigns to the nearest cluster [12]. Assume we have N entities, $e_1, \ldots, e_N$ and k clusters, $C_1, \ldots, C_k$ in the corpus. Then, algorithm repeatedly computes distance from centroids m_j and assigns e_i to the nearest cluster C_j. Then, it recomputes centroid $m_1, \ldots, m_k$ with new cluster members until it converges or reaches maximum iterations. The main goal is to minimize the distance between documents in the same cluster. It is the most popular algorithm by its simplicity. However, the convergence rate depends on initial centroid nodes.

On other hands, **Spectral Bisection (SB)** depends on the global connectivity between input nodes and repeatedly divide into two groups using the corresponding eigenvector of the second smallest eigenvalue [12]. Assume we have a similarity graph G, then we construct a laplacian graph $L = G - D$ and computes the second smallest eigenvalue λ_2 and corresponding eigenvector v_2. Then, we divide nodes into two groups based on the sign of the corresponding components in v_2. Then, it repeatedly applies the same algorithm to the subgraphs until it reaches a reasonably small size. It generally shows good performance when the clusters are evenly divided with overhead to compute eigenvector at each iteration.

Similarity Propagation (SP) [2] takes measures of similarity between pairs of nodes and simultaneously considers all nodes as potential centroids. At each iteration, similarity values (responsibility and availability) are exchanged between nodes until a high-quality set of centroids and corresponding clusters are gradually emerged. Responsibility $R(\vec{v}_{e_p}, \vec{v}_{e_q})$ indicates how well suited $\vec{v}_{e_q}$ is to work as a centroid for $\vec{v}_{e_p}$, considering other centroids for $\vec{v}_{e_p}$. In contrast, availability $A(\vec{v}_{e_p}, \vec{v}_{e_q})$ reflects how appropriate it would be for $\vec{v}_{e_p}$ to choose $\vec{v}_{e_q}$ as its centroid vector. The responsibility and availability are iteratively computed by

$$R(\vec{v}_{e_p}, \vec{v}_{e_q}) = G(\vec{v}_{e_p}, \vec{v}_{e_q}) - \max_{\vec{v}'_{e_q} \neq \vec{v}_{e_q}} \{A(\vec{v}_{e_p}, \vec{v}'_{e_q}) + G(\vec{v}_{e_p}, \vec{v}'_{e_q})\} \quad (1)$$

$$A(\vec{v}_{e_p}, \vec{v}_{e_q}) = \min\{0, R(\vec{v}_{e_q}, \vec{v}_{e_q}) + \sum_{\vec{v}'_{e_p} \in \{\vec{v}_{e_p}, \vec{v}_{e_q}\}} \max\{0, R(\vec{v}'_{e_p}, \vec{v}_{e_q})\}\} \quad (2)$$

where G is a similarity graph.

Latent Dirichlet Allocation (LDA) [3] is a Bayesian network that clusters web pages using a mixture of topics. As shown in Figure 2, for each web page d, a multinomial distribution θ over topics is randomly sampled from a Dirichlet with parameter α. Then, to generate each word, a topic z is chosen from this topic distribution, and a word w is generated by randomly sampling from a topic-specific multinomial distribution ϕ_z. To estimate θ and ϕ, we used collapsed Gibbs sampling approach as follows:

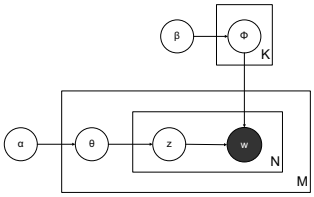

Figure 2. Latent Dirichlet Allocation.

$$P(z_i = k|z_i, w, \alpha, \beta) \propto \frac{\beta_v + n_{k,v}}{\sum_{v=1}^{V} \beta_v + n_{k,v}} \cdot \frac{\alpha_k + n_{m,k}}{\sum_{k=1}^{K} \alpha_k + n_{m,k}} \quad (3)$$

where K is the number of topics, and V is the number of words. Intuitively, the probability is proportional to the product of the probability of words in the given documents and the probability of topics in the given words.

III. Experimental Validation

For experiments, we used 12 name data sets in [1], [15]. It has twelve personnel names and each name set has different number of categories. The cluster size distribution is also extremely skewed and biased. For example, *Cheyer* has 97 documents with two classes but only one document belongs to one class and the other class has 96 documents. *Kaebling* shows the similar characteristic. Figure 3 shows the characteristics of the name data set. Each color shows different classes and the sizes of color bar shows the number of documents belong to the cluster. The number of classes in the name data varies a lot and the class size distributions are extremely skewed.

We constructed a term document matrix A with normalization and stemming options using TMG [13] software. We also deleted commonly used terms from the terminology dictionary. Term frequency (TF) and inverse document frequency (IDF) are used for local and global term weighting, respectively. Then, we constructed a document-document similarity graph matrix $G = A^T \times A$. Intuitively, the element $A(i, j)$ shows the similarity between documents d_i and d_j. If two documents share a lot of terminologies, then $A(i, j)$ has big magnitudes. On the other hands, if two documents do not share terminologies, then $A(i, j)$ stores zero value. Finally, to remove weakly connected relationships, we used a filtering by dropping insignificant values. We used 0.3 as our threshold.

To compare the performance, we measured RAND index which has been used in statistics community to compare two clustering results [16]. Given a set of n documents $D = \{d_1, \ldots, d_n\}$, we wish to compare two clusters: C and S. The resulting cluster of our algorithm is defined as $C = \{C_1, \ldots, C_k\}$ and manually derived solution set is defined as cluster $S = \{S_1, \ldots, S_k\}$. Then, RAND index is defined as

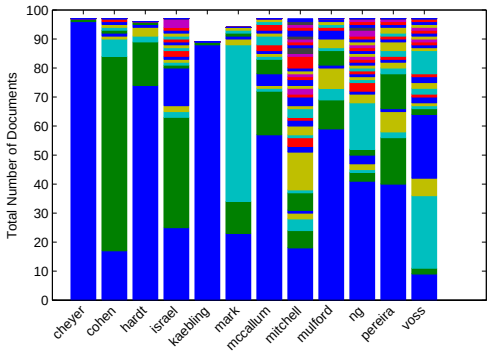

Figure 3. Problem Statistics. Each color represent different clusters.

$$RAND\ index = \frac{a+b}{a+b+c+d} \quad (4)$$

where a, b are agreement and c,d are disagreement between C and S. RAND index has a value between 0 and 1 with 0 indicating that two data clusters do not agree on any pair of points and 1 indicating that two data clusters are exactly the same.

Table I shows that numClus+K-means, numClus+SB or numClus+SP generate similar accuracies, and numClus+K-means improves about 16% accuracy of cluster results over HAC. However, numClus+LDA shows the poorest method because most name data sets have a number of clusters (e.g., K=37 and N=92 in the Tom Mitchell name data set) and several Tom Mitchells have similar occupations that share similar topics which degraded LDA performance. Interestingly, in some data sets (e.g., Cheyer and Kaelbling data sets), where there are only two clusters and one cluster has a majority of web documents, LDA outperforms the other methods. For instance, the *rand index* of HAC, numClus+K-means, numClus+SB, numClus+SP, and **numClus+LDA** are $0.11, 0.27, 0.12, 0.17$, and 0.38, respectively in the Cheyer data set as shown in Figure 4.

We also measured a relative error which is the ratio of difference between predicted and ground truth number of clusters. As illustrated in Figure 1, *the number of clusters approximated by numClus ($R_{numClus}$)* is considerably close to ground truth, while there exist big gaps between *the number of clusters generated by HAC (R_{HAC})* and ground truth. Indeed, the average relative error between $R_{numClus}$ and ground truth is ten times less than that of between R_{HAC} and ground truth.

IV. Related Works

Bekkerman et. al. proposed two algorithms to disambiguate web appearances of people in a social network in their paper [1]. One is based on link structure of web pages and another is using multi-way distributional clustering

Name	HAC	nClus+KM	nClus+SB	nClus+SP	nClus+LDA
Cheyer	0.11	0.27	0.12	0.17	0.38
Cohen	0.42	0.46	0.45	0.46	0.40
Hardt	0.41	0.41	0.59	0.43	0.32
Israel	0.76	0.72	0.74	0.70	0.35
Kaebling	0.03	0.59	0.49	0.50	0.53
Mark	0.60	0.65	0.58	0.64	0.23
McCallum	0.65	0.66	0.66	0.67	0.62
Mitchell	0.94	0.94	0.92	0.88	0.26
Mulford	0.65	0.64	0.63	0.65	0.12
Ng	0.67	0.83	0.83	0.82	0.03
Pereira	0.67	0.76	0.79	0.74	0.70
Voss	0.86	0.84	0.84	0.92	0.43
Average	**0.56**	**0.65**	**0.64**	**0.62**	**0.36**

Table I
RESULTS OF UNSUPERVISED METHODS.

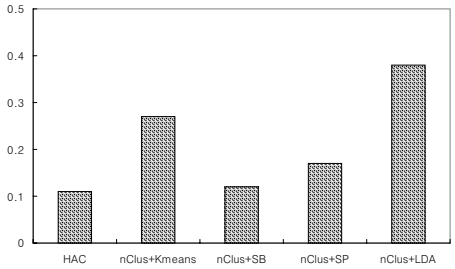

Figure 4. Rand Index for Cheyer name data set.

method. Their algorithms show improvement in the aspect of accuracy. Minkov et. al. used lazy graph walk algorithm to disambiguate names in email documents in their paper [6]. They provided a framework for email data, where content, social networks and a timeline to integrated in a structured graph. Banerjee et. al. proposed multi-way clustering on relation graphs in [7]. Different types of entities are simultaneously clustered based not only on their intrinsic attribute values but also on multiple relations between entities. On et al. introduced multi-level graph partitioning scheme to address the scalable issue of name disambiguation problem on both bibliographic and information retrieval domains [8].

On the other hand, to estimate the number of clusters, a number of approaches have been proposed. In particular, recent leading studies are Gap Statistic [11] and Clest [10]. However, such methods have been developed to address the problem in which the cluster sizes are well distributed. In addition, they focus on estimating a small number of clusters (e.g., at most 2 or 3 clusters). In authors awareness, this is the first paper to provide a systematic approach in solving a mixed entity resolution problem. In addition, we analyzed the name disambiguation problem characteristics, proposed an unsupervised clustering algorithm to solve extremely skewed clustering problem. Our experiment results show some promising in choosing a proper algorithm based on the problem and computational environments.

V. CONCLUSION

In this paper, we proposed an unsupervised clustering algorithm to solve mixed entity problem. Experimental results show an improvement of about 16% accuracy of clustering results at best case. We also proposed an algorithm to estimate the number of clusters, named *numClus*, using a document similarity graph. Based on our experiment, *numClus* identified the number of clusters ten times more correctly than Hierarchical Clustering.

Scalability is a huge problem in large data set such as Web pages. To provide a scalable algorithm, we are working on unsupervised clustering methods based on multi-level graph partitioning approach. In addition, it is infeasible for every clustering methods to correctly (perfectly) cluster web pages at all. To cope with this practical issue, we will apply the concept of feedback and investigate semi-clustering problem (induced by users' feedback) in our future work.

REFERENCES

[1] R. Bekkerman and A. McCallum. "Disambiguating Web Appearacnes of People in a Social Network". WWW 2005.

[2] B. Frey and D. Dueck. "Clustering by Passing Messages Between Data Points". J. Science, vol. 315, 2007.

[3] F. Chua. "Dimensionality Reduction and Clustering of Text Documents". Technical Report, Singapore Management University, 2009.

[4] I. Lee, B. On, and S. Yoon. "Algebraic Algorithms to Solve Name Disambiguation Problem". Int'l Conf. on Data Mining, Las Vegas, USA, July, 2009.

[5] D. Lee, B. On, J. Kang, and S. Park. "Effective and Scalable Solutions for Mixed and Split Citation Problems in Digital Libraries". ACM SIGMOD Workshop on Information Quality in Information Systems (IQIS), Baltimore, MD, USA, June, 2005.

[6] E. Monkov, W. Cohen, and A. Ng. "Contextual Search and Name Disambiguation in Email using Graphs". SIGIR'06.

[7] A. Banerjee, S. Basu, and S. Merugu. "Multi-way Clustering on Relation Graphs". SIAM Data Mining'07.

[8] B. On, and D. Lee. "Scalable Name Disambiguation using Multi-level Graph Partition". SIAM Data Mining'07.

[9] L. Kaufman and P.Rousseeuw. "Finding Groups in Data: An Introduction to Cluster Analysis" Wiley, 1990.

[10] S. Dudoit, and J. Fridlyand. "A Prediction-based Resampling Method for Estimating the Number of Clusters in a Dataset". Genome Biology 3(7), 2002.

[11] R. Tibshirani, G. Walther, and T. Hastie. "Estimating the Number of Clusters in a Dataset via the Gap Statistic". J. R. Statist. Soc. B(2001) 63, Part 2, pp. 411-423.

[12] J. Han, M. Kamber, and A. Tung. "Spatial clustering methods in data mining: A survey". Taylor and Francis, 2001.

[13] D. Zeimpekis and E. Gallopoulos. "TMG: A MATLAB toolbox for generating term document matrices from text collections". 2006.

[14] Jain, Murty, and Flynn. "Data clustering: A review". ACM Computing Surveys, 31, 1999.

[15] R. Bekkerman. "Name Data Set". "$http : //www.cs.umass.edu/ronb$".

[16] W. Rand. "Objective criteria for the evaluation of clustering methods". Journal of the American Statistical Association, Vol. 66, pp. 846-850, 1971.