IEICE TRANS. INF. & SYST., VOL.E93–D, NO.11 NOVEMBER 2010

LETTER

A Data Cleansing Method for Clustering Large-Scale Transaction Databases*

Woong-Kee LOH[†a)], **Yang-Sae MOON**[††], *Members*, *and* **Jun-Gyu KANG**[†††], *Nonmember*

SUMMARY In this paper, we emphasize the need for data cleansing when clustering large-scale transaction databases and propose a new data cleansing method that improves clustering quality and performance. We evaluate our data cleansing method through a series of experiments. As a result, the clustering quality and performance were significantly improved by up to 165% and 330%, respectively.

key words: *clustering, data cleansing, large-scale transaction databases*

1. Introduction

Data mining has been pursued since the 1990's, and clustering is an important technique in data mining. Clustering is finding the groups of objects having similar features, and it has been rigorously studied [1] since it has a wide range of applications. An example of such applications is target marketing. It is finding groups of customers having similar purchasing patterns and then establishing marketing strategies according to the patterns.

Recently, transaction databases have become a new target of clustering [1], [3]. A *transaction* is defined as a set of related items, and a *transaction database* is a database consisting of the transactions obtained in an application [6], [7]. As an example, Figure 1 shows four transactions in a transaction database in the application of search engine services. Each transaction contains the search keywords issued in the same user's session.

Transaction databases have introduced a few technical challenges. First, the objects handled in previous clustering algorithms were represented as d-dimensional vectors [2]. That is, they were represented as the points in d-dimensional space and were processed based on the Euclidean distance between them [3]. However, the transactions in transaction databases cannot be represented as d-dimensional vectors; they are called *categorical data* [3]. Second, the size of transaction databases is much larger than the dataset handled in previous algorithms [5]. While the size of datasets in previous algorithms is about several KBs to several MBs, transaction databases have sizes of several GBs up to several TBs.

USERID=37264:
amusement park, cherry blossom, mall of america, entrance fee, disneyland
USERID=93272:
freeway, traffic condition, shortcut
USERID=20438:
media player, skins, lyric words, download
USERID=72620:
major league, ichiro, baseball cap

Fig. 1 An example of transaction database.

In this paper, we emphasize the need for data cleansing, which is a pre-processing step before clustering on transaction databases, and propose a new data cleansing method that improves clustering performance and quality. Previous clustering algorithms did not consider the data cleansing process. In fact, transaction databases, such as search keyword databases, contain a lot of noise. For example, there are meaningless search keywords such as 'tjdnfeorhddnjs' that never appear again in the database. This sort of noise causes an increase of the number of useless clusters and the degradation of clustering performance and quality.

2. Related Work

Most of previous clustering algorithms handled only data objects that can be represented as d-dimensional vectors. There are small number of clustering algorithms that handle categorical data or transaction databases, and the most representative one is the ROCK algorithm [3]. It was shown in [3] that we could only get unsatisfactory clustering result on categorical data based on the Euclidean distance. Therefore, ROCK adopted Jaccard coefficient as a similarity measure between categorical data. However, since ROCK has the time complexity higher than $O(n^2)$, where n is the number of objects, it can hardly be applied to large-scale transaction databases.

Efficient clustering algorithms on large-scale transaction databases have been proposed in [6], [7]. A new notion of *large item* has been proposed in [6]. For a pre-specified support $\theta(0 \leq \theta \leq 1)$ and a transaction item e, if the ratio of clusters containing e in a cluster C_i is larger than θ, the item e is defined as a large item in the cluster C_i; otherwise, it is defined as a *small item*. The clustering algorithm in [6], which we call the *LARGE* algorithm in this paper, is

Manuscript received April 28, 2010.
Manuscript revised June 25, 2010.

[†]The author is with the Department of Multimedia, Sungkyul University, Korea.
[††]The author is with the Department of Computer Science, Kangwon National University, Korea.
[†††]The author is with the Department of Industrial & Management Engineering, Sungkyul University, Korea.

*This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2008-331-D00487).

a) E-mail: woong@sungkyul.ac.kr

DOI: 10.1587/transinf.E93.D.3120

executed in the direction of maximizing the number of large items and simultaneously minimizing small items by trying to bring the same transaction items together in a cluster.

The CLOPE algorithm [7], an improvement of LARGE, is also a heuristic algorithm and maximizes clustering quality by iteration. The algorithm does not use the notion of large/small items; it proposed a more efficient measure for computing clustering quality. CLOPE algorithm was shown in [7] to have better clustering performance and quality than ROCK and LARGE through a series of experiments.

The problems of LARGE and CLOPE are as follows. The algorithms did not consider the effect of noise data and assumed that the number of result clusters k is very small. However, in actual transaction databases, there contained a lot of noise data with very low frequencies, and the number of result clusters is fairly close to the number of transactions n. As a matter of fact, k should be highly variable depending on transactions in the database and items contained in the transactions. If k is very small compared with n, the average number of transactions in a cluster should be very high, and such large clusters should have little practical usefulness. LARGE and CLOPE have the time complexity of $O(nk)$, which approaches $O(n^2)$ as k approaches n.

In a broad sense, a text database or a document database can be regarded as a form of transaction database; a term and a document correspond to an item and a transaction, respectively. However, these databases have a few essential differences from the transaction databases. We omit the explanation on the differences due to page limitation; please refer to [4] for detailed explanation.

3. Data Cleansing

In this section, we explain the need for data cleansing and propose a new data cleansing method that improves clustering performance and quality. Our data cleansing method decides the usefulness of items according to their frequencies in transactions. Figure 2 shows the item frequencies in two real-world transaction databases. The horizontal axis represents item frequencies, and the vertical axis represents the number of items. As shown in the figure, there exist a lot of items whose frequencies are very small. The two transaction databases are explained in detail in Sect. 4.

Transaction items with too low or too high frequencies have negative effects on clustering performance and quality. We explain the phenomenon with examples. We use the same similarity measure between transactions as ROCK as the following Eq. (1):

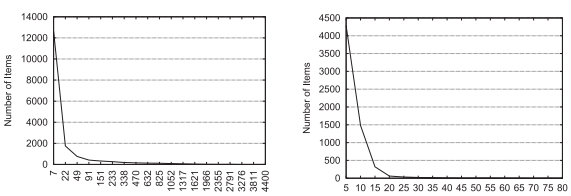
$$sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}, \tag{1}$$

where the denominator represents the number of whole items (without duplication) contained in transactions T_1 and T_2, and the numerator represents the number of items commonly contained in T_1 and T_2.

First, we explain the effect of the items with too low frequencies. Assume that similarity threshold θ between transactions is given as $\theta = 0.5$. Consider three transactions $T_1 = \{a, b, c, x, y, z\}$, $T_2 = \{b, c, d, p, q, r\}$, and $T_3 = \{a, c, d, s, t, u, v, w\}$. Then, for every transaction pair T_i and T_j ($i \neq j, 1 \leq i, j \leq 3$), it holds that $sim(T_i, T_j) < \theta$, and hence the transactions T_1, T_2, and T_3 does not form a cluster. However, by removing the items with very low frequencies (i.e., $x, y, z, p, q, r, s, t, u, v, w$), T_1, T_2, and T_3 become $T'_1 = \{a, b, c\}$, $T'_2 = \{b, c, d\}$, and $T'_3 = \{a, c, d\}$, respectively. Since, for every transaction pair T'_i and T'_j, it holds that $sim(T'_i, T'_j) \geq \theta$, three transactions T'_1, T'_2, and T'_3 should form a useful cluster. In fact, we can easily find enormous number of such transactions as T_1, T_2, and T_3 in real-world transaction databases. The problem due to low frequency items cannot be solved by adjusting or lowering the threshold θ, because the number of low frequency items is not constant across transactions and hence the threshold cannot be fixed.

Second, we show an example where clustering quality is degraded due to the items with too high frequencies. Consider four transactions $T_1 = \{a, b, c, d, x, y\}$, $T_2 = \{c, d, x, y, z, w\}$, $T_3 = \{q, r, x, y, z, w\}$, and $T_4 = \{o, p, q, r, z, w\}$. Since, for every transaction pair T_i and T_j ($1 \leq i, j \leq 4$), it holds that $sim(T_i, T_j) \geq \theta$, it is highly likely that the transactions T_1, T_2, T_3, and T_4 should form a large useless cluster $C_L = \{T_1, T_2, T_3, T_4\}$. However, by removing the items with very high frequencies (i.e., x, y, z, w), T_1, T_2, T_3, and T_4 become $T'_1 = \{a, b, c, d\}$, $T'_2 = \{c, d\}$, $T'_3 = \{q, r\}$, and $T'_4 = \{o, p, q, r\}$, respectively. The transactions T'_1, T'_2, T'_3, and T'_4 naturally form two useful clusters $C_1 = \{T'_1, T'_2\}$ and $C_2 = \{T'_3, T'_4\}$. Similarly to low frequency items, there are enormous number of transactions such as T_1, T_2, T_3, and T_4 in real-world transaction databases, and the problem due to high frequency items cannot be solved by adjusting or raising the threshold θ.

We assume that the item frequency shown in Fig. 2 should follow the lognormal or the exponential distribution. Based on this assumption, our data cleansing method performs as the following. First, in the transaction database, we count the number of transaction items for each item frequency (a positive integer value). Next, using the (item frequency, count) pairs, we estimate the parameters such as mean μ and standard deviation σ for the lognormal or the exponential distribution. Finally, for a pre-specified parameter s, we remove all the items whose frequencies are either less than $(\mu - s\sigma)$ or greater than $(\mu + s\sigma)$. After removing



(a) AOL database. (b) Keywords database.

Fig. 2 Item frequencies in two real-world transaction databases.

such items, we also remove empty transactions whose items have been entirely removed. In most cases, s should be $3 \sim 5$.

In the case of lognormal distribution, the estimates for two parameters μ and σ are obtained using the following Eq. (2):

$$\hat{\mu} = \frac{\Sigma_{i=1..n} \ln x_i}{n} \ , \ \hat{\sigma}^2 = \frac{\Sigma_{i=1..n} (\ln x_i - \hat{\mu})^2}{n} \ , \tag{2}$$

where n is the number of transaction items, and x_i represents item frequency. If there are k items whose frequencies are x_i, then x_i appears k times in Eq. (2).

In the case of exponential distribution, we compute the estimates for two parameters μ and σ using the following Eq. (3):

$$\hat{\mu} = \frac{1}{\hat{\lambda}} \ , \ \hat{\sigma}^2 = \frac{1}{\hat{\lambda}^2} \ , \tag{3}$$

where the estimate $\hat{\lambda}$ is computed as the following:

$$\hat{\lambda} = \frac{1}{\bar{x}} \ , \ \bar{x} = \frac{1}{n} \Sigma_{i=1..n} x_i \ . \tag{4}$$

Choosing which of two distributions for a specific transaction database is highly dependent on human expert's view. In our experiments, while choosing any of two distributions contributed to the improvement of clustering quality and performance, the lognormal distribution was more effective. Moreover, improper selection of parameter s value could result in worse clustering performance and quality. Larger s values were advantageous for the lognormal distribution, while smaller s values were advantageous for the exponential distribution.

Our data cleansing method can improve the quality of *incomplete* clustering results. CLOPE cannot always achieve complete clustering; actually, in most cases, its clustering results are incomplete. In such cases, our method helps improve clustering quality as well as clustering performance.

4. Evaluation

In this section, we evaluate our data cleansing method through a series of experiments. For our evaluation, we implemented CLOPE [7] and compared clustering quality and performance between two cases: case (1) using our data cleansing method and case (2) without using it. In case (1), the target transaction databases are pre-processed by our data cleansing method and then clustered by CLOPE, while, in case (2), the databases are directly clustered by CLOPE.

As explained in Sect. 2, CLOPE is a heuristic algorithm that enhances clustering quality by iteration. The algorithm computes a quality measure called *profit* of the intermediate clustering result at every iteration, and it stops when the profit does not increase any more. In our evaluation, we use the final profit as the clustering quality measure. For an intermediate or final clustering result $C = \{C_1, \ldots, C_k\}$, profit is defined as the following [7]:

$$Profit_r(C) = \frac{\sum_{i=1}^{k} \frac{S(C_i)}{W(C_i)^r} \times |C_i|}{\sum_{i=1}^{k} |C_i|} \ , \tag{5}$$

where $S(C_i)$, $W(C_i)$, and $|C_i|$ represent the numbers of entire items, distinct items, and transactions in a cluster C_i, respectively, and $r(> 0)$ represents *repulsion*, which is an input parameter for adjusting inter-cluster similarity; higher repulsion implies tighter similarity.

It was justified experimentally in [7] that, by using the profit as a metric of clustering quality, CLOPE was more effective than the previous algorithms. In the experiment, CLOPE was run on the mushroom dataset which contains human classification information on poisonous and edible mushrooms. CLOPE achieved the accuracy of 100% for the repulsion $r \geq 3.1$.

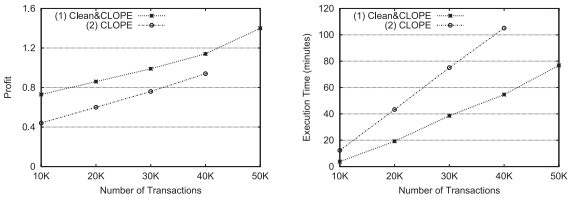
We used two datasets for our evaluation: (a) AOL search query database and (b) keyword registration database. The AOL database consists of about 20 M queries issued by about 650 K users from March 1 through May 31, 2006. The database is a list of records, and every record consists of five fields *AnonID*, *Query*, *QueryTime*, *ItemRank*, and *ClickURL*. The first three fields AnonID, Query, and QueryTime represent anonymous user ID, search keyword by the user, and timestamp when the query was issued, respectively. The fields ItemRank and ClickURL are optional, and they appear when the user clicked on any item in query result; they represent the rank and URL of the item clicked by the user, respectively. The keyword registration database is a transaction database; each transaction consists of a URL and a list of registered keywords. The same keyword can be registered by multiple URLs. When a query on a certain keyword is issued, the URLs that registered the keyword are shown in the query result.

We transformed AOL database into a transaction database in the form shown in Fig. 1 for clustering by CLOPE. Since a record in AOL database has a query at one time, a user's search queries are spread into multiple records, which appear adjacently in the AOL database. The queries by the same user are collected to form a record in the transaction database.

We used the user-id field (AnonID) when transforming AOL dataset into a transaction database. A transaction in the transaction database shown in Fig. 1 contains all the query terms of the same user-id. The query terms of the same user-id are collected into one transaction, and different transactions have different user-ids. Hence, the inter-transaction similarity based on user-id becomes always zero. We believe that the recommender systems should undergo similar procedures.

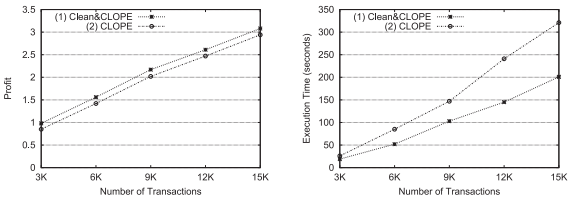
We set repulsion for CLOPE as $r = 1.5$, which is a largest value permitted by our system. We assumed that the number of transaction items follow the lognormal distribution and set $s = 5$.

Figure 3 shows the result of the first experiment using (a) AOL database; it compares clustering quality and performance between cases (1) and (2). In case (2), for the number of transactions 50 K, our program was terminated abnor-



(a) Clustering quality. (b) Clustering performance.

Fig. 3 Comparison of clustering quality and performance using AOL database.



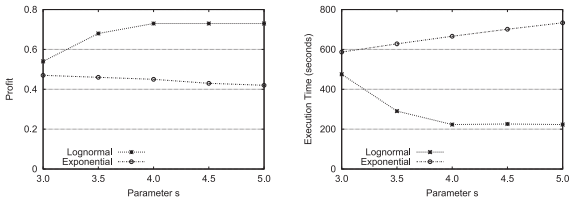
(a) Clustering quality. (b) Clustering performance.

Fig. 4 Comparison of clustering quality and performance using keyword database.

mally, which is most likely due to lack of main memory and swap space. As shown in the figure, clustering quality and performance was improved by applying our data cleansing method for every number of transactions. The improvement ratio of quality and performance reached up to 165% and 330%, respectively. In case (1), a much smaller number k of clusters were formed by CLOPE under the same settings. For that reason, since CLOPE has $O(nk)$ time complexity, we could gain the improvement of clustering performance.

We performed the second experiment using (b) keyword database with the same settings as the first experiment, and the result is shown in Fig. 4. As in Fig. 3, clustering quality and performance was also improved by applying our data cleansing method for every number of transactions. The improvement ratio of quality and performance reached up to 115% and 166%, respectively.

The third experiment was performed for two distributions and a few parameter s values. We used (a) AOL database used in the first experiment, and the number of transactions was set as 10 K. The experimental result is shown in Fig. 5. With the lognormal distribution, clustering quality and performance converge to a point for s values larger than or equal to 4.0. This means that there is no improvement in clustering quality and performance by our data cleansing method. With the exponential distribution, smaller s values were advantageous for improving clustering quality and performance.



(a) Clustering quality. (b) Clustering performance.

Fig. 5 Comparison of clustering quality and performance for different distributions and s values.

The time for data cleansing is not included in the graphs in Figs. 3, 4, and 5. Our data cleansing method needed as small time as less than a second in our experiments, which is almost negligible compared with the time for clustering.

References

[1] F. Beil, M. Ester, and X. Xu, "Frequent term-based text clustering," Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD), pp.436–442, Edmonton, Alberta, Canada, July 2002.

[2] Z. Feng, B. Zhou, and J. Shen, "A parallel hierarchical clustering algorithm for PCs cluster system," Neurocomputing, vol.70, no.4-6, pp.809–818, Jan. 2007.

[3] S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," Information Systems, vol.25, no.5, pp.345–366, 2000.

[4] W.-K. Loh, Y.-S. Moon, and J.-G. Kang, "A data cleansing method for clustering large-scale transaction databases," Computing Research Repository (CoRR), 1004.4718v1, April 2010.

[5] A. Nanopoulos and Y. Manolopoulos, "Efficient similarity search for market basket data," VLDB Journal, vol.11, no.2, pp.138–152, 2002.

[6] K. Wang, C. Xu, and B. Liu, "Clustering transactions using large items," Proc. Int'l Conf. Information and Knowledge Management (CIKM), pp.483–490, Kansas City, Missouri, USA, Nov. 1999.

[7] Y. Yang, X. Guan, and J. You, "CLOPE: A fast and effective clustering algorithm for transactional data," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp.682–687, Edmonton, Alberta, Canada, July 2002.