APPLYING INFORMATION RETRIEVAL FOR MARKET BASKET RECOMMENDER SYSTEMS

Tapio Pitkaranta
Helsinki University of Technology
Department of Computer Science and Engineering
Finland

Keywords: Information retrieval, Similarity search, Coded data sets, Nearest neighbor, Kernel functions.

Abstract: Coded data sets form the basis for many well known applications from healthcare prospective payment system to recommender systems in online shopping. Previous studies on coded data sets have introduced methods for the analysis of rather small data sets. This study proposes applying information retrieval methods for enabling high performance analysis of data masses that scale beyond traditional approaches. An essential component in today's data warehouses to which coded data sets are collected is a database management system (DBMS). This study presents experimental results how information retrieval indexes scale and outperform common database schemas with a leading commercial DBMS engine in analysis of coded data sets. The results show that flexible analysis of hundreds of millions of coded data sets is possible with a regular desktop hardware.

1 INTRODUCTION

This study[1] focuses on coded data sets [2] that are used as compact representations of real world processes. Coded data sets have well known domains such as minimum data sets in healthcare that have been used as representations of patient care processes for over one hundred years (WHO, 2004). Today minimum data sets constitute the basis for the healthcare payment systems throughout the world. Other well known domains for coded data sets include market basket analysis, that have been used to build different sorts of recommendation systems for online shopping.

Until now, traditional retail business has not been able to utilize the information it constantly collects to *directly* support its primary business. In the online world, however, it is possible even to modify the outline of an online store on-the-fly based on what we know from a particular customer. Yes, *Customers who bought this book, also bought, Users who liked this movie, also liked...* etc. Such modularity on the store outline is not easy to achieve in many traditional business sectors. For instance, modifying the outline of a supermarket on-the-fly as a single customer collects items into his/her market basket does not seem possible. But why are we lacking even simple recommendation systems in traditional business sectors?

In this paper we consider methods for customer recommendation systems in traditional business sectors. In this context large masses of imprecise data are available. We argue that using those data masses for direct recommendations would be the modern way of feeding back relevant aggregations from data collected by business processes in order to boost the business processes in real-time fashion. This already happens in the online world.

The rest of this paper is organized as follows. Section 2 presents the technical problem, previous studies and background for this study. Section 3 presents an information retrieval model for efficient analysis of coded data sets. Section 4 presents an experimental evaluation of the proposed model. Section 5 presents the conclusions and discussion.

2 BACKGROUND

In coded data sets each value belongs to a coding scheme, coding system, concept hierarchy or classifi-

[1] This work was partially supported by the Academy of Finland and the Emil Aaltonen's Foundation

[2] For this study the classifications are significant. In a way *classified* data sets would emphasize their role more accurately. However, we use the term *coded* because *classified* is often used to refer to something confidential or secret, and that is not the case here.

Pitkaranta T. (2009).
APPLYING INFORMATION RETRIEVAL FOR MARKET BASKET RECOMMENDER SYSTEMS.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Databases and Information Systems Integration*, pages 138-143
DOI: 10.5220/0001991101380143

cations. These coding systems are essentially discrete finite sets of possible values. Each value in the coding scheme has some kind of corresponding activity or item in the real world. One coded data set can represent a particular primary process such as one purchase by a customer or one delivery from a vendor. In a typical case the coded values belong to a mono dimensional multi-hierarchical product catalog. However, there are various other types of coding schemes varying from multi dimensional keyword based categorizations to complete ontologies.

2.1 Data Space

We should have a look at the data space characteristics because of the impreciseness of the data. The number of all possible coded data sets is the number of code combinations. The number of combinations C can be calculated with Equation 1. In Equation 1, r is the number of different coding schemes in the data space, the *SCHEME* denotes the number of codes in a single coding scheme and *CODES* denotes the maximum number of coded values that belong to this coding scheme in one data set:

$$C = \prod_{i=1}^{r} \binom{SCHEME_i}{CODES_i} \qquad (1)$$

Using Equation 1 with an example number of 100 coded values and corresponding coding scheme with 10000 codes we get: $C = \prod_{i=1}^{r} \binom{SCHEME_i}{CODES_i} = \binom{10000}{100} = 6.5 \cdot 10^{241}$ combinations. As we can see from Equation 1, even with reasonably small number of codes and possibilities, the coded data sets constitute a somewhat high dimensional data space. The data space is also sparse since there are a relatively small number of real business processes compared to the theoretical number of different data set combinations.

2.2 Number of Data Sets

The type and amount of coded data sets to be analyzed, depends on the type of the primary business process that the data represents. Data can be collected from various time intervals from short periods to even dozens of years. Therefore, the amount of data can vary from a couple of coded data sets to practically infinite. It should be also noted, that the granularity of the coding scheme used for describing organizations primary business processes has an impact on how much information is stored and how precisely the coded data represents the actual primary business processes.

As an example case we could use retail business where we have a large supermarket chain with a nationwide coverage. If we have 100 million citizens who on average visit a supermarket once a week, we get 5 billion yearly visits. Each visit to a supermarket can be represented with one coded data set. If we assume 20% market share for this large scale supermarket chain, we can expect one billion coded data sets to be stored into their information systems on a yearly basis. If we assume that the data is stored for ten years, we get total history of 10 billion coded data sets.

2.3 Related Work

The analysis of coded data sets has been approached from different angles from data mining, association mining, knowledge discovery, collaborative filtering, inductive databases, neural networks, etc. Collaborative filtering systems attempt to offer personalized recommendations of items based on information about similarities among user's tastes. One of the most well known collaborative filtering systems are Tapestry, the Ringo system and Video Recommender. For a comprehensive overview on such systems readers are referred to (Herlocker et al., 2004).

Within retail business, the market baskets have been widely studied using association rules (Agrawal and Srikant, 1994). An example is the PROFSET model, that is proposed for analyzing the business value of a single product based on the cross-selling effects it has with other products (Brijs et al., 2000). Previous studies on efficient similarity searches for market baskets have proposed exact methods such as signature-bases representation for coded data sets. Example is named S^3B, however the scheme was tested in the experiments with only a fairly small databases with 100k transactions (Nanopoulos and Manolopoulos, 2002). Furthermore, considering the impreciseness and inaccuracy of the data, applying exact methods is somewhat questionable although it enables a compact representation of the data sets. There are also several studies on read intensive databases such as column oriented database structures (Stonebraker et al., 2005; Harizopoulos et al., 2006) and integration of information retrieval and database systems (Roelleke et al., 2008; Grabs et al., 2001).

2.4 Analysis of Coded Data Sets

Most of the commercial database management systems (DBMS) and data warehouses store data in with the relational databases model that was introduced almost 40 years ago. For coded data sets, possible relational structures are the normalized and de-normalized database schemas. In the de-normalized schema, one single row in a database table represents one (coded) data set. In the normalized schema the dimensions are represented with separate tables that are linked to the fact table. Using these schemas, we are able to implement a recommendation system using traditional SQL. However, in both database schemas we encounter some challenges. Using the normalized schema, we have to obtain data from three different tables in the query. With the de-normalized schema the table consumes significant amount of space due to data anomalies. Second, in traditional SQL we cannot directly calculate distribution of product codes since they reside in different columns. Number of indexes we need and the number of *AND* / *OR* conditions that appear in the queries are other problems that occur because of having all codes in different columns.

Other problems with using traditional SQL is to benefit from the underlying multi-hierarchical coding scheme. How to integrate implement on the fly aggregation and ranking mechanism with SQL? Probably the recommendation system should be able to create associations not based on particular cigarette and alcohol labels but rather certain aggregations or higher level product groups. This will also make the SQL queries complex.

3 KERNEL METHODS AND INFORMATION RETRIEVAL

Kernel methods are classes of algorithms that are used for pattern analysis. The general task of pattern analysis is to find and study general types of relations in general types of data. The types of relations can include clusters, principal components and correlations. The types of data can be text documents, images, sequences, etc.

Kernel methods approach this problem by projecting the data into a high dimensional feature space, in which each coordinate corresponds to one feature of the data items. In this space, a variety of methods can be used to find relationships in the data. The so called *kernel trick* enables kernel methods to operate in the feature space without ever calculating the coordinates of the data in that space, but rather computing the distance with different similarity functions in the feature space. This operation is often computationally cheaper and non-separable data set can become separable after the projection.

Kernel-based algorithms such as term vector analysis are used in high dimensional data spaces for calculating distances of two data sets. Frequently used techniques for locating nearest neighbors are distance measurements such as the Cosine Angle Distance (CAD) and Euclidean distance (EUD) (Zobel and Moffat, 2006). These distance measurements have been reported to perform similarly in high dimensional data spaces for nearest neighbor queries (Gang et al., 2004).

Because of the flexibility requirements for the analysis of the imprecise coded data sets and high dimensional domain data space, this study proposes using the vector space model analyzing existing coded data set databases. The proposed model is referred as *information retrieval model* as the applied kernel techniques are commonly used in IR (Zobel and Moffat, 2006).

The coded data sets are interlinked by the coding scheme. The similarity between the query vector $Q = (q_1, q_2, ..., q_t)$ and document representing the coded data set $D_i = (d_{i1}, d_{i2}, ..., d_{it})$ using corresponding query weights q_j for each term, is described by Equation 2:

$$s(q, d_i) = \frac{\sum_{j=1}^{t} (q_j d_{i,j})}{\sqrt{\sum_{j=1}^{t} q_j^2}\sqrt{\sum_{j=1}^{t} d_{i,j}^2}} \tag{2}$$

A common way for defining the document weights is described in Equation 3 and query weights in Equation 4 (Zobel and Moffat, 2006; Wilkinson and Hingston, 1991; Haykin, 1999). In these equations the $log(N/f_j)$ is the so-called *inverse document frequency* where N is the number of documents in the database and f_j is the number of documents that contain term t_j. Furthermore, the tf_{ij} is the *within document frequency* indicating the number of occurrences of term t_j in document i.

$$d_{ij} = tf_{ij} \cdot \log(\frac{N}{f_j}) \tag{3}$$

$$q_j = \begin{cases} \log \frac{N}{f_j} & \text{, if term } t_j \text{ appears in the query;} \\ 0 & \text{, otherwise.} \end{cases} \tag{4}$$

This technique is flexible for weighting different parts of the query vector which is appropriate in the domain: some codes are more important than the others. This also provides us flexible ranking if we want to get not only exact matches, but rather data sets that are somewhat close. We can also apply the underlying coding schemes to empower the query. While searching for exact matches with the exact codes, we can also use codes that are *nearby* based on the coding scheme structure.

4 EXPERIMENTAL EVALUATION

This section presents experimental evaluation of the analysis of coded data sets using the proposed information retrieval model. An essential component in vast majority of commercial data warehousing and B2B applications is a DBMS engine. Therefore the proposed model is benchmarked against a leading commercial DBMS engine with two relational database schemas suitable for coded data sets.

The use case is a recommendation system for a large scale retail business with possibly hundreds of millions of coded data sets. Therefore the three implemented models are referred in this experiment as recommenders.

Synthetic coded data sets were used to evaluate the performance and accuracy of the recommenders over a large range of data characteristics. The content of the coded data sets corresponding to the coding scheme were created using a method that was presented in (Agrawal and Srikant, 1994). Table 1 lists the materials that were used in this experiment with the corresponding initialization parameters. As can be seen from Table 1, the number of coded data sets varied from 10 thousand to 100 million and the size of the material from 3.1 MB to 30.2 GB. Each coded data set contained varying number of codes and the total number of codes in the materials was over one billion.

Table 1: Materials used in the experiments.

Name	$\lvert T \rvert$	$\lvert I \rvert$	$\lvert D \rvert$	Size (MB)
T10I10D10K	10	10	10000	3.1
T10I10D100K	10	10	100000	31.4
T10I10D1000K	10	10	1000000	322
T10I10D10000K	10	10	10000000	3 217
T10I10D10000K	10	10	100000000	30 192

The coding scheme was created with a hierarchical and mono-dimensional structure that contains four hierarchy levels each with maximum number of 10 child nodes on each hierarchy level. Together the coding scheme contained 10 thousand codes.

4.1 Implementation

The materials listed in Table 1 were inserted into a leading commercial DBMS under two different database schemas and to the implementation of the proposed information retrieval model. The database schemas used in this experiment were the de-normalized and normalized schema.

The benchmarks were implemented using Java programming language using a range of Open Source components. The recommender systems were implemented using object oriented programming. All common components were re-used, and stub-recommender that uses the common components was created for eliminating performance issues that are not related to the actual recommender core components. Web-based User interface was built for viewing data sets interactively. The user interface was implemented using Java Enterprise Edition and a Java Enterprise Application Server. The database was accessed using Java Database Connectivity (JDBC) API.

The hardware used for the performance tests was a regular desktop pc with 2.0 GHZ dual core Intel processor and with 2 GB of physical RAM memory and a 160 GB hard drive. The experiments were executed on Windows Vista platform running Java(TM) SE Runtime Environment version 1.6.0.06.

4.2 Benchmarking

Important criteria for a recommendation system are the accuracy of the recommendation and performance with large number of data sets, various search patterns and simultaneous users.

In the recommendation experiment, typical patterns were used to evaluate the accuracy of the recommendation. As the materials used in this study are synthetic frequent patterns and characteristics are known. In the recommendation accuracy experiment was conducted by removing one code from a commonly occurring pattern and letting the recommender fill the missing value for that particular code. The original code was used as a golden standard against which the recommended code was compared.

In the performance benchmarking both frequently and infrequently occurring patterns were used. In all experiments 50 different patterns were used. These 50 different patterns were searched 100 times and each experiment was executed 10 independent times. The results represent average values from these all executions.

The parameters for the recommenders were *kNN*, $topN$ and the $|D|$. The *kNN* refers to *k nearest neighbours* that defines the number of neighbours from which the recommendation will be decided from. The $topN$ refers to the number of recommendations that the recommender should provide for the users in a descending order. The $|D|$ refers to the generated data set that are listed in Table 1.

4.3 Results

The performance of different recommenders with frequent and infrequently occurring patterns are de-

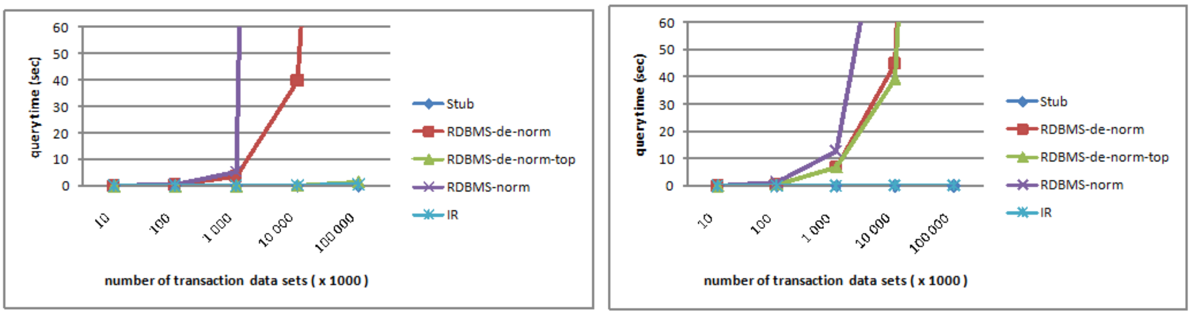

Figure 1: Query speed using frequent and infrequent patterns (kN=5, topN=5)

picted in Figure 1. Five different recommenders were implemented: 1) *Stub* recommender for eliminating non recommendation core related performance issues 2) *RDBMS-de-norm* used the de-normalized database schema 3) *RDBMS-de-norm-top* used the de-normalized database schema returning only topN results 4) *RDBMS-norm* used the normalized database schema and 5) *IR* that used the proposed information retrieval model.

The variable in Figure 1 is the number of data sets $|D|$ in the materials that are listed in Table 1. The variables *kNN* and *topN* were fixed: $kNN = 5$ and $topN = 5$. Performance with frequently occurring patterns is depicted in the left hand side of Figure 1 and infrequent patterns on the right. As can be seen Figure 1, with frequent patterns the de-normalized relational data schema using top *kNN* queries performs reasonably well although it is clearly outperformed by the information retrieval recommender. With frequent occurring patterns and a fairly small *kNN*, the de-normalized relational schema does not need to scan through the whole database in order to retrieve the requested *kNN* neighbors. This is because no ranking mechanism is implemented to this recommender. As can be seen from Figure 1, the normalized relational schema has performance issues with smaller data sets as de-normalized schema due to the time consuming processing of three separate database table.

From the right hand side of Figure 1 it can noted that recommending with infrequent data sets cause performance problems for all recommender except the information retrieval recommender. For infrequent data sets RDBMS engine needs to scan through large portion of the database before being able to retrieve the requested *kNN*.

The recommendation accuracy for different recommendation core components are depicted in Figure 2 using *topN* as a variable on the left and *kNN* in the middle and right. In this experiment, the recommendation was based on the frequency distribution of codes occurring in the *kNN*. From the left hand side of Figure 2 we can see that recommendation accuracy of the relational models increases as *kNN* is incremented. This is because the recommender component does not have a ranking mechanism that would return nothing but exactly matching results. Therefore, increasing *kNN* does not reflect the accuracy after all matching neighbors have been located.

The information retrieval model, however, is able to return accurate results with a fairly small *kNN* as the ranking mechanism and similarity search provides the *kNN* with a greater accuracy than pure boolean algebra. The information retrieval recommender suffers from increasing the *kNN* over certain point. That is because the ranking mechanism is able to return also data sets that are nearby although they are not exact matches. This is both a pro and a con. As depicted in the right of Figure 2, the accuracy of the relational recommendation models suffer from a rapid decline as infrequent or partially invalid noisy patterns are used: there are no exactly matching *kNN* for those patterns. This could be the case if the data warehouse is not up-to-date for instance with all new product codes and customer purchases. Using pure relational algebra appears to provide poor results.

5 CONCLUSIONS AND DISCUSSION

This study proposed using information retrieval based model for efficient analysis of coded data sets. The model uses the vector space model to represents coded data sets, such as patient care in healthcare or customer purchases in retail business. Different similarity functions enable flexible and scalable analysis of large code data set masses.

Requirement for the analysis model was high scalability to hundreds of millions of data sets as the use-case was large scale retail business recommender system. This study showed that building such a recommender system is possible using regular desktop hard-

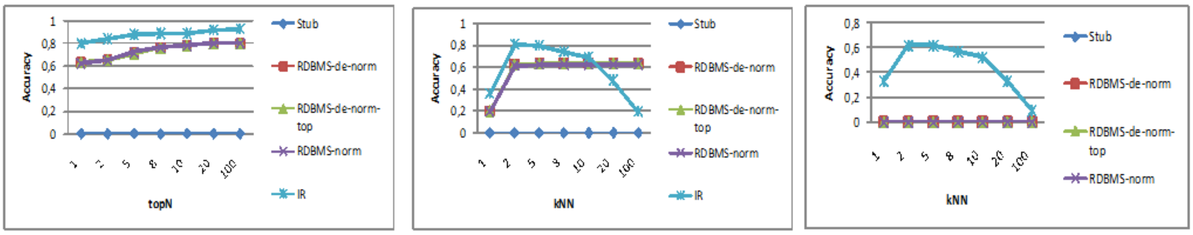

Figure 2: Recommendation accuracy using varying *topN* (left) and *kNN* (middle) with frequent patterns. Recommendation accuracy using infrequent noisy patterns with varying *kNN* on the right.

ware. This study presented experimental comparison between the proposed information retrieval model and a leading commercial DBMS system with two relational database schemas. Based on these experiments, the information retrieval model outperforms the relational schemas in both performance and flexibility.

The experimental results show that the pattern analysis performance using relational database schemas and a regular desktop computer begins to have problems with several million coded data sets. The results also show that the information retrieval model is able to produce rapid results to flexible queries with 100 million coded data sets with regular desktop hardware.

Overall the results show that the information retrieval recommender provides more accurate recommendation with a small amount of *kNN*. The model is able to produce recommendations also with noisy and partially invalid patterns which is often the case in the real world as the quality of the data is far from perfect.

The proposed model utilizes kernel methods and vector distance functions for efficient nearest neighbor queries. Different types of distance functions enable flexibility for different use cases. In the future we are applying the model to healthcare data sets. This includes empowering the similarity functions with domain specific knowledge from the healthcare coding schemes.

REFERENCES

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Brijs, T., Goethals, B., Swinnen, G., Vanhoof, K., and Wets, G. (2000). A data mining framework for optimal product selection in retail supermarket data: the generalized profset model. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 300–304, New York, NY, USA. ACM.

Gang, Q., Sural, S., Gu, Y., and Pramanik, S. (2004). Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237. Michigan State University, ACM. ISBN: 1-58113-812-1.

Grabs, T., Böhm, K., and Schek, H.-J. (2001). Powerdb-ir: information retrieval on top of a database cluster. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 411–418, New York, NY, USA. ACM.

Harizopoulos, S., Liang, V., Abadi, D. J., and Madden, S. (2006). Performance tradeoffs in read-optimized databases. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 487–498. VLDB Endowment.

Haykin, S. (1999). *Neural Networks - A Comprehensive Foundation*. Prentice Hall. ISBN: 0-13-273350-1.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53.

Nanopoulos, A. and Manolopoulos, Y. (2002). Efficient similarity search for market basket data. *The VLDB Journal*, 11(2):138–152.

Roelleke, T., Wu, H., Wang, J., and Azzam, H. (2008). Modelling retrieval models in a probabilistic relational algebra with a new operator: the relational Bayes. *VLDB Journal: Very Large Data Bases*, 17(1):5–37.

Stonebraker, M., Abadi, D. J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., O'Neil, E., O'Neil, P., Rasin, A., Tran, N., and Zdonik, S. (2005). C-store: a column-oriented dbms. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 553–564. VLDB Endowment.

WHO (2004). *International Statistical Classification of Diseases and Related Health Problems, Instruction manual*, volume 2. World Health Organization. ISBN: 92 4 154649 8.

Wilkinson, R. and Hingston, P. (1991). Using the cosine measure in a neural network for document retrieval. *ACM*, pages 202–210.

Zobel, J. and Moffat, A. (2006). Inverted files for text search engines. *ACM Comput. Surv.*, 38(2):6.