SLICE: A Novel Method to Find Local Linear Correlations by Constructing Hyperplanes*

Liang Tang[1], Changjie Tang[1], Lei Duan[1], Yexi Jiang[1], Jie Zuo[1,**], and Jun Zhu[2]

[1] School of Computer Science, Sichuan University
610065 Chengdu, China
{tangliang,tangchangjie}@cs.scu.edu.cn
[2] National Center for Birth Defects Monitoring
610041 Chengdu, China

Abstract. Finding linear correlations in dataset is an important data mining task, which can be widely applied in the real world. Existing correlation clustering methods may miss some correlations when instances are sparsely distributed. Other recent studies are limited to find the primary linear correlation of the dataset. This paper develops a novel approach to seek multiple local linear correlations in dataset. Extensive experiments show that this approach is effective and efficient to find the linear correlations in data subsets.

Keywords: Data Mining, Linear Correlation, Principal Component Analysis.

1 Introduction

Linear correlations reveal the linear dependencies among several features in a dataset. Finding theses correlations is an interesting research topic. For example, in sensor network, a latent pattern, which is the linear correlation of multiple time series data received, can be used to detect the data evolution [8]. Principal component analysis (PCA) is able to capture linear correlations in a dataset [4]. PCA assumes that all instances in a dataset are in the same correlation. However, instances collected from the real world may have different characteristics, so the linear dependencies among features may be different in different data subsets.

Some clustering methods are developed to find data clusters in subspaces [2, 3, 5, 9]. Correlation clustering methods, such as 4C, try to seek clusters in a linear correlation [6, 7]. The first step of these methods is to generate ε–neighborhoods first. However, it is hard to generate high quality ε–neighborhoods to find linear correlations, when the distribution of instances in the correlations is sparse. In [1], a method called CARE is proposed to find local linear correlations. This method adopts PCA to analyze the linear correlations on both feature subsets and instance subsets. CARE focuses on finding the primary linear correlations from the whole dataset. It is not suitable to find linear correlations that exist in data subsets.

* This work was supported by the National Natural Science Foundation of China under grant No. 60773169 and the 11th Five Years Key Programs for Sci. &Tech. Development of China under grant No. 2006BAI05A01.
** Correspondence author.

Q. Li et al. (Eds.): APWeb/WAIM 2009, LNCS 5446, pp. 635–640, 2009.

The study of this paper focuses on finding multiple linear correlations in data subsets. We refer such correlations only existed in the subset of dataset as *local linear correlations*. The main challenge for us is that the number of data subsets is so large that it is hard to enumerate all subsets in reasonable runtime. Thus, we design a method to search linear correlations by constructing hyperplanes. The time complexity of our proposed method is polynomial.

Our Contributions. (1) analyzing the limitations of applying current methods on finding linear correlations in data subsets; (2) developing a heuristic algorithm SLICE (**s**ignificant **l**ocal l**i**near **c**orrelation s**e**arching) to find multiple local linear correlations in data subsets. The basic idea of our method is using a heuristic to construct hyperplanes that represent linear correlations; (3) conducting extensive experiments to show that SLICE is effective to find correct correlations in both synthetic and real-world datasets.

2 Significant Local Linear Correlation Searching

Firstly, we use the definition of strongly correlated feature subset proposed in [1] to measures both accuracy and significance of the local correlation.

Definition 1 (Significant Local Linear Correlation). Given a data matrix D containing M tuples, each tuple has d features. Let C_D be the covariance matrix of D, and $\{\lambda_i\}$ $(1 \le i \le d)$ be the eigenvalues of C_D, where $\lambda_1 \le \lambda_2 \le \ldots \le \lambda_d$, The data subset $S = \{x_{j1}, \ldots, x_{jm}\}$ $(0 \le j_t \le M, t = 1, 2, \ldots, m)$ is in a *significant local linear correlation* if following conditions are true:

$$f(S,k) = \sum_{t=1}^{k} \lambda_t / \sum_{t=1}^{N} \lambda_t \le \varepsilon \tag{1}$$

$$m / M \ge \delta \tag{2}$$

where ε, δ and k $(k \le .N)$ are user defined parameters. The meanings of these user defined parameters are as same as in [1]. The computation cost would be very large if each subset is enumerated and tested to make sure it is in a significant local linear correlation or not. Given a data matrix with M tuples, there are total 2^M data subsets to be tested, which is impractical for real-world applications.

Our SLICE (**s**ignificant **l**ocal l**i**near **c**orrelation s**e**arching) adopts a heuristic to find the data subset within a significant local linear correlation. Suppose the dataset has M tuples and d features, the heuristic searching begins with an initial seed (d tuples), and let S be the set of the d tuples, then S absorbs the next "best" tuple in rest tuples iteratively until the value of $f(S, k)$ becomes larger than ε, which is introduced in Definition 1. If $|S|$ satisfies the requirement in Definition 1, the linear correlation established on S can be regarded as a *significant local linear correlation*. The concept of the "best" of a tuple will be discussed later in this section. We introduce the definition of distance from a tuple to a hyperplane firstly.

Definition 2 (Distance from a tuple to a hyperplane). Given a d dimensional data subset S and a tuple x. The distance from x to the hyperplane established on S, denoted as $d(x, S)$, is defined as follows.

$$d(x,S) = f(S \cup \{x\},k) - f(S,k) \tag{3}$$

where $f(S, k)$ is defined in Definition 1.

According to Definition 1 and Definition 2, we can see that if the distance from a tuple to the hyperplane is small, the increase of $f(S, k)$ would be small after S absorbs this tuple. So, in the process of searching, the tuple with the minimal distance to the hyperplane is the "best" tuple to be absorbed. Our approach to find the "best" tuple is efficient and incremental. The computational complexity of updating the covariance matrix of S in each iteration is $O(d^2)$ (d is the dimensionality). Considering the computational complexity of PCA, the time complexity of finding the minimum distance between a tuple and the hyperplane is $O(d^2+d^3)$. Thus, finding the "best" tuple in each step costs $O(n(d^2+d^3))$ at most.

The second key point of SLICE is how to arrange the initial seeds to start the searching to find all significant local linear correlations. If the number of features is d and the size of dataset is M, there are C_M^d different initial seeds to be tested. It is easy to see that the computational cost is large when the dimensionality is large. Fortunately, we can use a pruning strategy to set the initial seeds for searching. Then the whole process can converge very fast. SLICE does not choose the tuples as initial seeds which have been absorbed by a hyperplane again. Algorithm 1 describes the implementation details of SLICE.

Algorithm 1. SLICE (D, ε, δ, k)

Input: D: a d-dimensional data set D, ε, δ, k: user defined parameters.
Output: SS: data subsets that are in a *significant local linear correlation*.
Begin
1 $M \leftarrow |D|$, $C \leftarrow D$, $SS \leftarrow \varnothing$
2 **while** $|C| > d$
3 $seed \leftarrow \varnothing$
4 **while** $|seed| < d$
5 $seed \leftarrow seed \cup \text{RANDOM}(C, d)$
6 **end**
7 $S \leftarrow \text{SEARCH}(D, seed, \varepsilon, \delta, k)$
8 **if** $|S| \geq M \cdot \delta$ **then**
9 $SS \leftarrow SS \cup \{S\}$
10 **end**
11 $C \leftarrow (C - S)$
12 **end**
End.

The time complexity of Algorithm 1 is $O(t \cdot n(d^2+d^3))$, where t is the times of invoking searching method. In the worst case, $t = n/d$. If ε is small, the number of tuples got in Subroutine SEARCH(D, $seed$, ε, δ, k) is small. In this case, t would be large. So we can see that t is associated with parameter ε. Our experimental results show that t is less than n in most cases.

3 Experimental Study

To evaluate the performance of SLICE, we test it on several synthetic datasets and a real world dataset. SLICE is implemented using Matlab 7.6. The experiments are performed on a 1.8GHz PC with 2G memory running Windows Server 2003 operating system. The characteristics of experimental datasets are presented as follows.

- **Synthetic datasets:** we generate 500 distinct datasets. These synthetic datasets are categorized to 5 different groups. For example, "D300F4C3" means each dataset in this group has 300 tuples with 4 features, and contains 3 predefined local linear correlations.
- **Real world dataset:** we use NBA statistics dataset[1] to test the performance of SLICE. We use all 458 players' statistical scores in season 2006..

In order to evaluate the effectiveness of algorithms, we conduct SLICE on these 500 synthetic datasets. We compare the discovered correlations with predefined correlations in datasets. If SLICE finds all predefined correlations, we mark this run as a *success*. We set k=1. The values of ε, δ and *success* rate[2] are list in Table 1. From Table 1, we can conclude that our SLICE has high probability to reach all linear correlations in these synthetic datasets.

Table 1. Results for testing on synthetic datasets

D300F3C3	D400F3C4	D600F3C4	D600F4C4	D500F4C5
ε =0.0006	ε =0.0006	ε =0.0006	ε =0.0001	ε =0.0001
δ =0.3	δ =0.2	δ =0.2	δ =0.2	δ =0.18
Rate=100%	Rate=95%	Rate=99%	Rate=97%	Rate=100%

Next, we compare SLICE with 4C and CARE, since 4C and CARE are mostly recent works close to ours to the best of our knowledge.

a) Comparison with algorithm 4C: 4C is a kind of correlation clustering algorithm [6]. This algorithm has to generate $\varepsilon-$ neighborhoods at first. In the synthetic datasets, each correlation intersects with another correlation. Therefore, for this kind of datasets, the generated $\varepsilon-$ neighborhoods contain tuples in different correlations that would mislead searching the correct correlation clusters. Due to the space limit, we only give the results in dataset D300F3C3 here (Figure 1). Similar results are found in other datasets.

b) Comparison with algorithm CARE: In this experiment, we use CARE to find linear correlations in tuple subset. Figure 2 illustrates the hyperplane found by CARE in dataset D300F3C3. Based on this result, we can see the main limitation of CARE is that it is only capable to find the primary linear correlation of the dataset. Due to the space limit, we only give the results in dataset D300F3C3 here (Figure 2). Similar results are found in other datasets.

[1] http://sports.espn.go.com/nba/statistics

[2] We define the *success* rate is the percent of SLICE finds all predefined correlations over 500 datasets.

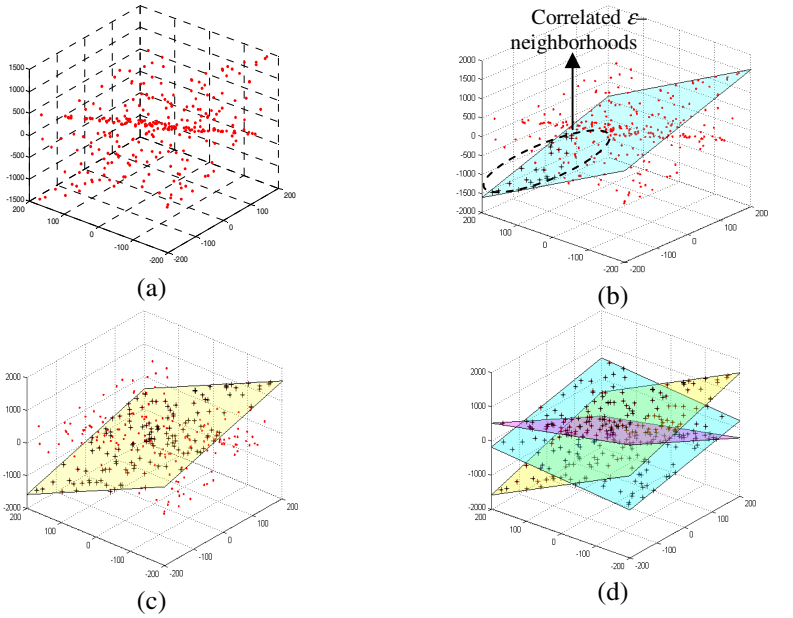

Fig. 1. **(a)** D300F3C3 dataset. **(b)** Correlation found by 4C in D300F3C3 dataset. **(c)** Correlation found by CARE in D300F3C3 dataset. **(d)** Correlations found by SLICE in D300F3C3 dataset.

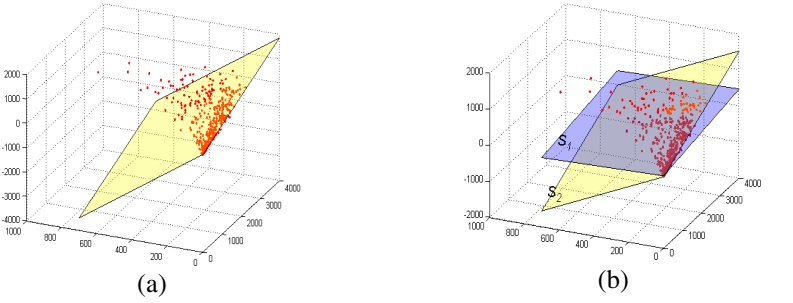

Fig. 2. **(a)** Correlations found by CARE in NBA dataset. **(b)** Correlations found by SLICE in NBA dataset.

As the authors of [1] had demonstrated that CARE can better results than 4C, we just compare our SLICE with CARE in NBA dataset. We use the same parameters in CARE and SLICE ($\varepsilon = 0.0006$, $\delta = 0.5$, $k = 1$). Table 2 lists the linear correlations discovered by CARE and SLICE. In common sense, different players have different

Table 2. *Success* rates of SLICE for each group of datasets

CARE	0.090645 * *minutes* – 0.976344 * *assists* – 0.196303 * *rebounds* = 0.796832
SLICE	0.157510 * *minutes* – 0.898574 * *assists* – 0.409580 * *rebounds* = 8.273140 0.112557 * *minutes* – 0.121851 * *assists* – 0.986146 * *rebounds* = -8.695694

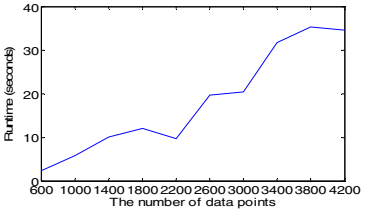

Fig. 3. Varying data size

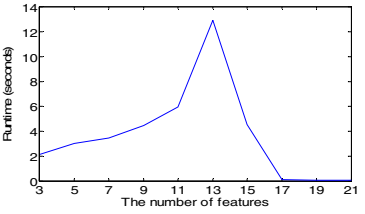

Fig. 4. Varying feature size

responsibilities in a match. We believe the results of SLICE are much closer to the real world compared the result of CARE.

In order to evaluate the efficiency and scalability of algorithms, we generate 10 datasets with different data sizes. Moreover, we generate 10 datasets with different dimensionalities to evaluate the scalability of SLICE on feature size. All these synthetic datasets has 4 predefined local linear correlations. The parameter $k = 1$, and $\varepsilon = 0.0001$, $\delta = 0.2$. The results are showed in Figure 3 and Figure 4.

4 Conclusions

Finding linear correlations in dataset has many real world applications. In this paper, we propose a method to find local linear correlations in data subsets. The full work of this paper could be seen in the place[3]. In future, we plan to integrate user interests in our method to find interesting local linear correlations. Furthermore, developing a method that can guarantee to find all correct linear correlations in polynomial time complexity in each execution is a challenging work.

References

1. Zhang, X., Pan, F., Wang, W.: CARE: Finding Local Linear Correlations in High Dimensional Data. In: The 24th IEEE International Conference on Data Engineering (ICDE), pp. 130–139 (2008)
2. Aggarwal, C., Yu, P.: Finding Generalized Projected Clusters in High Dimensional Spaces. In: ACM SIGMOD 2000, pp. 70–81 (2000)
3. Aggarwal, C., Wolf, J., Yu, P.: Fast Algorithms for Projected Clustering. In: ACM SIGMOD 1999, pp. 61–72 (1999)
4. Jolliffe, I.: Principal Component Analysis. Sprinter, New York (1986)
5. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In: ACM SIGMOD 1998, pp. 94–105 (1998)
6. Bohm, C., Kailing, K., Kroger, P., Zimek, A.: Computing Clusters of Correlation Connected Objects. In: ACM SIGMOD 2004, pp. 455–466 (2004)
7. Achtert, E., Bohm, C., Kriegel, H.-P., Kroger, P., Zimek, A.: Deriving Quantitative Models for Correlation Clusters. In: ACM KDD 2006, pp. 4–13 (2006)
8. Papadimitriou, S., Sun, J., Faloutsos, C.: Streaming Pattern Discovery in Multiple Time-Series. In: VLDB 2005, pp. 497–708 (2005)
9. Chakrabarti, K., Mehrotra, S.: Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Spaces. In: VLDB 2000, pp. 89–100 (2000)

[3] http://cs.scu.edu.cn/~tangliang/papers/slice_tangliang_waim09.pdf