4S: Scalable Subspace Search Scheme Overcoming Traditional Apriori Processing

Hoang Vu Nguyen[○] Emmanuel Müller[○●] Klemens Böhm[○]

[○]*Karlsruhe Institute of Technology (KIT), Germany*
{*hoang.nguyen, emmanuel.mueller, klemens.boehm*}*@kit.edu*

[●]*University of Antwerp, Belgium*
emmanuel.mueller@ua.ac.be

***Abstract*—In many real-world applications, data is collected in multi-dimensional spaces. However, not all dimensions are relevant for data analysis. Instead, interesting knowledge is hidden in correlated subsets of dimensions (i.e., subspaces of the original space). Detecting these correlated subspaces independent of the underlying mining task is an open research problem. It is challenging due to the exponential search space. Existing methods have tried to tackle this by utilizing Apriori search schemes. However, they show poor scalability and miss high quality subspaces.**

This paper features a *scalable subspace search scheme (4S)*, which overcomes the efficiency problem by departing from the traditional levelwise search. We propose a new generalized notion of correlated subspaces which gives way to transforming the search space to a correlation graph of dimensions. Then we perform a direct mining of correlated subspaces in the graph. Finally, we merge subspaces based on the MDL principle and obtain high dimensional subspaces with minimal redundancy. We theoretically show that our search scheme is more general than existing search schemes and has a significantly lower runtime complexity. Our experiments reveal that 4S scales near-linearly with both database size and dimensionality, and produces higher quality subspaces than state-of-the-art methods.

I. Introduction

The notion of correlation is one of the key elements of statistics and is important for many areas of applied science. For instance, correlations have just recently been exploited in entropy-based dependency analysis to identify novel structures in global health and human gut microbiota data [27]. In data mining, correlations of dimensions often indicate the existence of patterns (e.g., clusters [7] and outliers [15]), and are thus very important for knowledge discovery. In multi-dimensional data, however, patterns are often obscured in the full-space due to the presence of noisy features. Instead, they can be found in (possibly overlapping) correlated subsets of dimensions, so-called correlated subspaces of the original data space. Mining such subspaces, commonly referred to as subspace search, is crucial to unravel interesting knowledge and to understand multi-dimensional data.

Example: The facility management of our university stores indicator values of buildings, such as electricity, heating, gas, and water consumption per hour. Each dimension is one of these indicators of a specific building. In such data, not all indicators of all buildings are correlated with each other. Instead, there are different subsets of correlated indicators, e.g., the heating indicators of office buildings, the ones of the Chemistry department, and so on. Overlap among subsets is possible since buildings can both be office buildings and belong to the Chemistry department. In practice, detecting subsets of correlated indicators is important for facility managers. This is because they can understand the energy consumption of the university better from such subsets. For instance, they can apply specialized data analytics on just those subsets to find clusters or anomalous measurements. An example would be an abnormally high heating consumption among the office buildings. Clearly, one cannot observe such patterns when indicators are uncorrelated or data is distributed randomly. Lastly, it is preferable to find subsets with as many correlated indicators as possible, i.e., high dimensional subspaces. Returning redundant lower-dimensional projections of the same subspace distracts users and misses the bigger picture.

We observe three open challenges that have to be tackled for scalable subspace search, in particular w.r.t. dimensionality of data. First, it is unclear how to decide which subspaces have high correlations. Existing methods [7], [14], [15], [23], using an Apriori-style search scheme, impose a restrictive monotonicity on the correlation model: A relevant subspace has to be relevant in all of its lower-dimensional projections. However, this is only for efficiency reasons and may cause poor results in terms of quality. Second, one needs a scalable search scheme to navigate through the huge search space. For a data set with 40 dimensions the total number of subspaces is 2^{40} (more than 1 trillion). Looking at databases in practice (e.g., our facility management stores 540 dimensions), the search space is astronomically large. Obviously, this makes brute-force search impractical. Apriori-style methods, though employing the monotonicity restriction, still suffer from poor efficiency due to (a) their expensive mining of correlated dimension pairs and (b) their level-by-level processing that generates a tremendous number of candidates. Third, the final set of subspaces must be free of redundancy, i.e., it must contain high dimensional subspaces rather than their fragments. Existing methods use the monotonicity restriction. Hence, they detect redundant subspaces of low quality which are projections of the same high dimensional subspace.

We address these challenges by proposing 4S. In general, we depart from the traditional Apriori search scheme and its monotonicity restriction. In particular, we make scalable subspace search feasible by creating a new generalized notion of correlated subspaces: We define a subspace to have a high correlation if its member dimensions are *all pairwise correlated*. We later establish a relationship between our notion and the well-known total correlation [7] and prove that our notion is more general than existing ones. That is, given the same correlation measure, all subspaces found by Apriori-style methods are also discovered by 4S. As a result, we expect 4S to discover subspaces missed by such methods.

4S starts exploring the search space by computing pairwise correlations of dimensions. To ensure scalability, we devise two new efficient computation methods for this task. One of these methods builds upon AMS Sketch [5]. To our knowledge, we are first to use this theory for efficient computation of pairwise correlations of continuous random variables.

Based on the pairwise correlations computed, we map the subspace search problem to efficient mining of maximal cliques in a correlation graph. Hence, we get rid of the levelwise search of Apriori-style methods and directly construct higher-dimensional subspaces by maximal clique mining. Due to this non-levelwise processing, 4S neither requires to compute correlations of each subspace candidate nor to check an excessive number of its lower-dimensional projections.

To address the fragmentation issue of high dimensional correlated subspaces, we transform the problem to an MDL-based merge scheme of subspaces and merge the detected subspaces accordingly. While MDL is an established notion for model selection, its deployment to subspace search is new.

Overall, unlike Apriori-style methods [7], [14], [15], [23], 4S can find high dimensional correlated subspaces, and provides both efficiency and quality due to our main contributions:

- A generalized notion of correlated subspaces, relaxing restrictions of traditional models.
- A scalable subspace search scheme, including efficient pairwise correlation computation and direct construction of high dimensional correlated subspaces.
- An MDL-based merge of subspaces to reconstruct fragmented subspaces and remove redundancy

II. Related Work

Feature selection. Related methods such as PCA and others [9], [17] select one subspace only. However, a dimension not relevant for one subspace may form a correlated subspace together with other dimensions. Thus, these simplified schemes likely miss important subspaces. Conversely, 4S aims at mining multiple possibly overlapping subspaces.

Subspace search for specific tasks. There exist subspace search methods designed specifically for tasks such as outlier detection [2], [16], [22], clustering [3], [31], [20], and classification [32], [12]. However, they are strongly coupled with the underlying tasks. For instance, supervised feature selection focuses on the correlation between each dimension and the class label, not the correlations among dimensions. 4S in turn is unsupervised, and further, not bound to any specific task.

General subspace search. [7], [14], [15], [23] are recent proposals to mine overlapping subspaces, abstracting from any concrete task. They explore the search space using an Apriori-style search. However, due to the monotonicity restriction, they detect only low dimensional subspaces. Such subspaces in turn likely are different projections of the same high dimensional correlated subspaces. This causes redundancy that is well-known for most subspace mining models [21], [20]. Besides, they suffer severe scalability issue due to their expensive mining of correlated dimension pairs, and their levelwise search scheme which generates very many candidate subspaces. In contrast, 4S aims at a novel scalable search scheme that departs from these drawbacks. Experiments show that 4S yields good results with much less runtime.

III. Preliminaries

Consider a database $\mathbf{DB}$ of size N and dimensionality D. The set of dimensions is denoted as the full-space $F = \{X_1, \ldots, X_D\}$. Each dimension X_i has a continuous domain $dom(X_i)$ and w.l.o.g, we assume $dom(X_i) = [-v, v] \subseteq \mathbb{R}$.

A subspace S is a non-empty subset of F. Its dimensionality is written as $|S|$. The subspace lattice of $\mathbf{DB}$ consists of $D-1$ layers $\{\mathcal{L}_i\}_{i=2}^{D}$. Single-dimensional subspaces are excluded since one is interested in correlations of two or more dimensions. Every layer $\mathcal{L}_i$ contains $\binom{D}{i}$ many subspaces, each having i dimensions.

We aim at mining subspaces across all lattice layers whose member dimensions are highly correlated. Note that the search space is huge. For a dataspace with D dimensions the total number of possible subspaces is $O(2^D)$. For one subspace, one needs $O(D \cdot N)$ time to process, e.g., to compute the correlation. An overall complexity of $O(D \cdot N \cdot 2^D)$ makes brute-force search impractical. Even more sophisticated search schemes have severe scalability problems (cf. Section III-A). Hence, we propose a new scalable solution (cf. Section III-B).

A. Existing Search Schemes

Existing methods explore the search space based on the Apriori principle (APR) using a correlation measure for subspace assessment. Here we choose the total correlation for illustration purposes.

Definition 1: The total correlation of $\{X_i\}_{i=1}^{d}$ *is:*

$$T(\{X_1, \ldots, X_d\}) = \sum_{i=2}^{d} H(X_i) - H(X_i | X_1, \ldots, X_{i-1})$$

where $H(\ldots)$ *is the Shannon (differential) entropy.*

The total correlation is used in [7], which computes entropies by estimating probability density functions through discretization. For APR, one can either keep a top number of subspaces at each layer (beam-based) or impose a threshold on the subspace correlation (threshold-based). Recently, [15], [23] point out that the beam-based scheme allows more intuitive parameterization than the threshold-based one. Thus, for better presentation, we stick to the former. However, our discussion is also applicable to the threshold-based scheme [7], [14].

APR starts at layer $\mathcal{L}_2$. For each layer $\mathcal{L}_i$ visited, APR computes the total correlation $T(S)$ for each candidate subspace $S \in \mathcal{L}_i$. The top MAX_NUM subspaces $\{S^i_{r(j)}\}_{j=1}^{MAX\_NUM}$ with the highest total correlation are selected. MAX_NUM is the beam size. $\{S^i_{r(j)}\}_{j=1}^{MAX\_NUM}$ are also used to determine which subspaces to examine in the next layer $\mathcal{L}_{i+1}$. In particular, a subspace S^{i+1} in $\mathcal{L}_{i+1}$ is considered iff all of its i-dimensional projections are in $\{S^i_{r(j)}\}_{j=1}^{MAX\_NUM}$. This is known as the monotonicity restriction, which causes redundant processing: To reach one subspace, one needs to generate and examine all of its lower-dimensional projections.

APR stops when either there is no more layer to explore or the set of candidate subspaces in the current layer is empty. Assume that MAX_NUM is set such that APR reaches layer $\mathcal{L}_k$. The time complexity of APR is $O(D^2 \Delta + 2^k \cdot \Delta \cdot MAX\_NUM)$ where the first term is the cost of exploring $\mathcal{L}_2$, the second term is the cost of exploring higher layers, and Δ is the cost of computing the correlation of each subspace. For instance, $\Delta = \Theta(N)$ in [7].

Since the monotonicity property imposes strict restriction on high-level layers (i.e., high k), APR tends not to reach high dimensional subspaces. To resolve the issue, MAX_NUM must be significantly large. However, this causes APR to process many candidate subspaces at each layer visited. Further, to process a subspace, APR requires to examine exponentially many lower-dimensional projections to ensure that they all have high correlation. These cause its runtime to become very high. Even when MAX_NUM is kept low, APR still suffers from poor scalability due to its expensive mining of $\mathcal{L}_2$. Further, setting MAX_NUM to low values fails to offset the monotonicity restriction. This prevents APR from discovering high dimensional subspaces. Only lower-dimensional fragments of correlated subspaces are detected. Thus, the quality of subspaces is impacted. In summary, APR (a) is inefficient, (b) tends to miss high dimensional correlated subspaces, and (c) fragments them into many redundant lower-dimensional subspaces.

B. Overview of 4S *Processing*

To avoid the exponential runtime w.r.t. the dimensionality, 4S does not explore the subspace lattice in a levelwise manner. Instead, 4S initially mines subspaces of high correlations in $\mathcal{L}_2$. They are then combined to directly create higher-dimensional subspaces. In short, 4S works in three steps. First, we compute the correlation of each pair of dimensions and only keep the top K pairs (i.e., subspaces of $\mathcal{L}_2$) with the largest correlations. Setting K is explained in Section V.

Next, we construct an undirected correlation graph $\mathcal{G}_\mathcal{D}$ representing our search space of subspaces. Its nodes are the dimensions, connected by an edge iff their correlation is in the top K values. Following our new notion of correlated subspaces, we mine maximal cliques of this correlation graph. They serve as candidate subspaces. The toy example in Figure 1 displays a correlation graph for a 10-dimensional data set. There are 45 possible subspaces in $\mathcal{L}_2$; $K = 10$ of which are picked to construct $\mathcal{G}_\mathcal{D}$. From $\mathcal{G}_\mathcal{D}$, 4S finds three maximal cliques (subspaces): $S_1 = \{1, 2, 3, 4\}$, $S_2 = \{1, 3, 4, 5\}$, and $S_3 = \{7, 8\}$.

Mining maximal cliques on $\mathcal{G}_\mathcal{D}$ may also produce subspaces that are projections of the same subspaces due to the restriction on pairwise correlations (i.e., through K). For instance, in Figure 1, dimension 5 is connected to all dimensions in S_1 except for dimension 2. This leads to the detection of two separate subspace fragments S_1 and S_2 that have high overlap with each other. It would make sense to merge S_1 and S_2 to create the larger subspace $\{1, 2, 3, 4, 5\}$. This also helps us to cope with real-world data where perfect pairwise correlation between dimensions of correlated subspaces may not always be fulfilled. Thus, we propose to merge similar subspaces using an MDL-based approach. Following this step, we obtain even higher-dimensional subspaces with minimal redundancy.

Overall, in contrast to APR, we can reach high dimensional correlated subspaces by our scalable search scheme, which consists of: (a) scalable computation of $\mathcal{L}_2$, (b) scalable mining of $\mathcal{L}_k$ with $k > 2$, and (c) subspace merge. While APR needs to impose the Apriori monotonicity restriction on all layers for efficiency purpose, we only require that dimensions of subspaces are pairwise correlated (i.e., restriction on $\mathcal{L}_2$). Next, we introduce each step of 4S in detail (cf. Section IV-VII). Further, we formally prove that 4S is more general than APR in Section V, and empirically show that 4S produces subspaces of higher quality than existing methods in Section VIII.

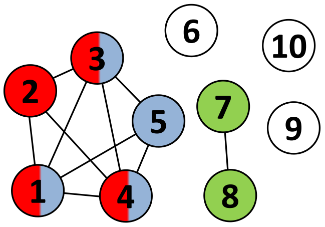

Fig. 1: Example of correlation graph.

IV. Scalable Computation of $\mathcal{L}_2$

Our goal is to have a correlation measure that captures both linear and non-linear correlation. The measure should also permit direct calculation on empirical data without having to estimate probability density functions or rely on discretization [27], [7]. Our research in turn is orthogonal to studying such a new measure, and we base ourselves on a simple yet effective correlation measure. It is defined as follows:

Definition 2: *The correlation of dimensions X and Y is:*

$$Corr(X, Y) = \int_{-v}^{v} \int_{-v}^{v} (F_{XY}(x, y) - F_X(x)F_Y(y))^2 dx dy$$

where $F(\ldots)$ is the cumulative distribution function (CDF).

$Corr$ has the nice property that it is defined based on CDFs which allow direct computation on empirical data. Let $\{x_i\}_{i=1}^N$ and $\{y_i\}_{i=1}^N$ be realizations of X and Y, respectively. The theorem below shows how to compute $Corr(X, Y)$ using these empirical values (cf. proof in [29]).

Theorem 1: *$Corr(X, Y)$ equals to*

$$\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(x_i, x_j))(v - \max(y_i, y_j))$$
$$-\frac{2}{N^3} \sum_{i=1}^{N} \left(\sum_{j=1}^{N} (v - \max(x_i, x_j)) \right) \left(\sum_{j=1}^{N} (v - \max(y_i, y_j)) \right)$$
$$+\frac{1}{N^4} \sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(x_i, x_j)) \sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(y_i, y_j))$$

Following Theorem 1, we need to compute three terms, referred to as T_1, T_2, and T_3, and $Corr(X, Y) = \frac{1}{N^2} T_1 - \frac{2}{N^3} T_2 + \frac{1}{N^4} T_3$. $Corr$ is based on [29]. However, we improve $Corr$ by devising approaches to efficiently compute it on large and multi-dimensional data sets. In particular, to compute $Corr(X, Y)$, we originally need $O(N^2)$ time. For D-dimensional data sets, the total runtime required to explore layer $\mathcal{L}_2$ becomes $O(D^2 N^2)$. This is a serious problem for the data sets under our consideration. To tackle the issue, we introduce two new approaches, *MultiPruning* and *Sketching*, to boost efficiency regarding both N and D. *MultiPruning* calculates the exact correlation. However, it still has issues regarding efficiency for large data sets. *Sketching* in turn trades accuracy for efficiency. Yet it is still better than APR (cf., Section VIII). Note that *Corr* deploys the same estimator as other quadratic measures of independence [25], such as [1], [30]. The difference only lies in different kernels employed. Thus, the ideas of *MultiPruning* and *Sketching* are also applicable to other measures of the same category. In other words, our method is not limited to only one correlation measure.

A. *MultiPruning*

MultiPruning aims at reducing the runtime by applying pruning rules for $Corr(X, Y)$ based on two upper bounds of T_1. It uses the fact that we only keep the top K pairs of dimensions with the largest correlation. Let $\{(x_{s(i)}, y_{s(i)})\}_{i=1}^N$ be $\{(x_i, y_i)\}_{i=1}^N$ sorted in descending order w.r.t. X. The upper bounds of T_1 are as follows.

Theorem 2: [CAUCHY-SCHWARZ BOUND]

$$T_1 \leq \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{N}(v - \max(x_i, x_j))^2 \sum_{j=1}^{N}(v - \max(y_i, y_j))^2}$$

Theorem 3: [SORTED-BASED BOUND] $T_1 \leq$

$$\sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)}) + 2\sum_{i=1}^{N}(v - x_{s(i)}) \sum_{j=i+1}^{N} (v - y_{s(j)})$$

The statistics required for the Cauchy-Schwarz bound, for instance $\sum_{j=1}^N (v - \max(x_i, x_j))^2$ for $1 \leq i \leq N$, can be pre-computed for each dimension in $O(N \log N)$ time. This is from our observation that $\sum_{j=1}^N (v - \max(x_i, x_j))^2 = \sum_{x_j \geq x_i} (v - x_j)^2 + \sum_{x_j < x_i} (v - x_i)^2$. That is, for each dimension, we first sort its data in descending order. Then, we loop through the data in that order and pre-compute the required statistics. To illustrate our idea, let us consider three data points $P_1 = (1, -1)$, $P_2 = (-1, 1)$, and $P_3 = (0, 0)$ (i.e., $dom(X) = [-1, 1]$). To compute the statistics for X, we sort X in descending order and obtain $\{1, 0, -1\}$. Then, we compute $\sum_{j=1}^3 (1 - \max(x_i, x_j))^2$ $(1 \leq i \leq 3)$ by looping through the sorted list once and obtain: 0 for P_1, 5 for P_2, and 2 for P_3. Similarly, for $\sum_{j=1}^3 (1 - \max(y_i, y_j))^2$ $(1 \leq i \leq 3)$, we obtain: 5 for P_1, 0 for P_2, and 2 for P_3. The statistics required to *exactly* compute the second term T_2 of $Corr(X, Y)$, which are $\sum_{j=1}^N (v - \max(x_i, x_j))$ for $1 \leq i \leq N$, can be pre-computed similarly. The statistic of the third term T_3, which is $\sum_{i=1}^N \sum_{j=1}^N (v - \max(x_i, x_j))$, is also computed during this phase by incrementally summing up the statistics of T_2 (per dimension).

During the pairwise correlation computation, we maintain the top K values seen so far. For a new pair of dimensions (X, Y), we first compute the bounds. This computation is in $O(N)$. Using the same example, we calculate the Cauchy-Schwarz bound by looping through the stored statistics once, and achieve: $\sqrt{0 \cdot 5} + \sqrt{5 \cdot 0} + \sqrt{2 \cdot 2} = 2$. Similarly, the exact value of the second term T_2 is computed. The sorted-based bound in Theorem 3 is obtained in $O(N)$ time as follows. We loop through the data sorted w.r.t. X. For each point $(x_{s(i)}, y_{s(i)})$, we compute $(v - x_{s(i)})(v - y_{s(i)})$ and $(v - x_{s(i)}) \sum_{j=i+1}^N (v - y_{s(j)})$ taking into account that $\sum_{j=i+1}^N (v - y_{s(j)}) = \sum_{j=1}^N (v - y_{s(j)}) - \sum_{j=1}^i (v - y_{s(j)})$. The sorted-based bound can also be computed w.r.t. Y. So in fact, we have two versions of this bound, one for X and one for Y. The exact value of T_3 is computed in just $O(1)$ time using its pre-computed statistics.

If any upper bound of $Corr(X, Y)$ is less than the K^{th} largest value so far, we can safely stop computing its actual value. Otherwise, we compute T_1, and hence $Corr(X, Y)$ (and update the top K correlation values), using Lemma 1.

Lemma 1: $T_1 = \sum_{i=1}^N (v - x_{s(i)})(v - y_{s(i)})$ $+2 \sum_{i=1}^N (v - x_{s(i)}) \sum_{j=i+1}^N (v - \max(y_{s(i)}, y_{s(j)}))$

That is, for each $x_{s(i)}$, we search for $y_{s(i)}$ in the list of values of Y sorted in descending order. For each value $y > y_{s(i)}$ encountered, we add $2(v - x_{s(i)})(v - y)$ to T_1. Once $y_{s(i)}$ is found, the search stops. Suppose that the position found is p, and the list has e elements. We add $2(e - p + 1)(v - x_{s(i)})(v - y_{s(i)})$ to T_1. We remove $y_{s(i)}$ from the list and proceed to $x_{s(i+1)}$. This helps us to avoid scanning the whole list and, hence, reduces the runtime. We note that $\sum_{i=1}^N (v - x_{s(i)})(v - y_{s(i)})$ is already computed during the sorted-based bound computation.

Computing $Corr(X, Y)$ for each pair of dimensions $\{X, Y\}$ costs $O(N^2)$. Thus, the worst-case complexity of *MultiPruning* is $O(D^2 N^2)$. However, our experiments show that *MultiPruning* is efficient in practice.

B. *Sketching*

To better address the scalability issue (i.e., quadratic in N), we propose *Sketching* as an alternative solution. First, we see that T_3 is computed in only $O(1)$ time using its pre-computed statistics. Thus, our main intuition is to convert the terms T_1 and T_2 to forms similar to that of T_3. We observe that T_1 and T_2 can be perceived as dot products of vectors. Such products can be efficiently estimated by AMS Sketch [5]. AMS Sketch provides rigorous theoretical bounds for its estimation and can outperform other sketching schemes [28]. However, to our knowledge, we are first to use this theory to efficiently compute pairwise correlations of continuous random variables. Our general idea is to use AMS Sketch to derive unbiased estimators of T_1 and T_2 that have forms similar to T_3. The estimators are unbiased since their expected values equal to their respective true values. To show that the estimation errors are small, we prove that the estimators concentrate closely enough around the true values of T_1 and T_2, respectively. Overall, *Sketching* reduces the time complexity of computing $Corr(X, Y)$ to $O(N \log N)$.

Sketching approximates $Corr(X, Y)$ through unbiased estimators by projecting X and Y onto random 4-wise independent vectors. Let $u, v \in \{\pm 1\}^N$ be two independent random 4-wise independent vectors. We estimate T_1 as:

Theorem 4: Let Z be a random variable that equals to

$$\sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(x_i, x_j))u_i v_j \sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(y_i, y_j))u_i v_j$$

then $E(Z) = T_1$ and $Var(Z) \leq 8[E(Z)]^2$.

Likewise, we estimate T_2 as:

Theorem 5: Let W be a random variable that equals to

$$\sum_{i=1}^{N}\sum_{j=1}^{N}(v-\max(x_i,x_j))u_i\sum_{i=1}^{N}\sum_{j=1}^{N}(v-\max(y_i,y_j))u_i$$

then $E(W) = T_2$ *and* $Var(W) \leq 2[E(W)]^2$.

We derive Theorems 4 and 5 using [5]. They allow us to approximate T_1 and T_2 by estimators having forms similar to that of T_3. Hence, $Corr(X, Y)$ can be approximated in $O(N \log N)$ time by pre-computing the statistics required in a way similar to *MultiPruning*. Please note that, we also need to ensure estimators to concentrate closely enough around their respective mean. To accomplish this, we apply Chebychev's inequality. For Z, its variance is upper-bounded by $8[E(Z)]^2$. By averaging over s_1 different values of u and v, the variance is reduced to at most $\frac{8[E(Z)]^2}{s_1}$. Using Chebychev's inequality, we have: $P(|Z - E(Z)| > \varepsilon E(Z)) \leq \frac{8}{s_1\varepsilon^2}$. If we repeat the averaging $s_2 = O(1/\delta)$ times and take the median of these averages, the relative error of Z w.r.t. $E(Z)$ is at most ε with probability at least $1 - \delta$, as proven in [5].

Similarly, by averaging over s_1 different values of u, the variance of W is reduced to at most $\frac{2[E(W)]^2}{s_1}$. Applying the Chebychev's inequality, we have: $P(|W - E(W)| > \varepsilon E(W)) \leq \frac{2}{s_1\varepsilon^2}$. We again boost the estimation accuracy by repeating the averaging $s_2 = O(1/\delta)$ times.

Sorting all dimensions costs $O(DN \log N)$. For each random vector and each dimension, it costs $O(N)$ to pre-compute the statistics. For all vectors and all dimensions, the total cost of pre-computing statistics is $O(s_1 s_2 DN)$. Since $s_1 s_2$ must be large enough to guarantee estimation accuracy, the cost of pre-computing statistics dominates that of data sorting. Computing the correlations for all dimension pairs and maintaining the top values cost $O(s_1 s_2 D^2 + D^2 \log K)$ with $O(s_1 s_2 D^2)$ dominating. Thus, the total time complexity of *Sketching* is $O(s_1 s_2 DN + s_1 s_2 D^2)$. In our experiments, $D < N$, i.e., the time complexity becomes $O(s_1 s_2 DN)$, a considerable improvement from $O(D^2 N^2)$. We note that the factor $s_1 s_2$ does not contribute much to the overall runtime and in practice *Sketching* scales linearly to both N and D.

V. Scalable Mining of $\mathcal{L}_k$

Based on the set of 2-dimensional subspaces found in $\mathcal{L}_2$, denoted as $\mathcal{S}_2$, we now explain how to mine subspaces in higher-level layers. According to our notion, a subspace has a high correlation if its member dimensions are all pairwise correlated. We now point out that subspaces fulfilling our notion likely have a high total correlation. We also formally prove that our new notion of correlated subspaces is more general than that of APR. That is, given the same correlation measure, all subspaces found by APR are also discovered by our mining scheme. Further, we will demonstrate empirically later on that, with our notion, 4S produces better subspaces than APR. First, let us consider a subspace S with all pairs $\{X_i, X_j\} \in \mathcal{S}_2$. W.l.o.g., assume that $S = \{X_1, \ldots, X_d\}$.

Lemma 2: The total correlation is lower-bounded by: $T(\{X_1, \ldots, X_d\}) \geq \sum_{i=2}^{d} H(X_i) - H(X_i|X_{i-1})$

Lemma 2 is derived from the fact that conditioning reduces entropy [24]. By definition, every pair $\{X_{i-1}, X_i\} \in \mathcal{S}_2$ has a high correlation. Following Definition 2, this means that $F(X_{i-1}, X_i)$ and $F(X_{i-1})F(X_i)$ deviate from each other. Thus, the joint density function $f(X_{i-1}, X_i)$ of X_{i-1} and X_i deviates from the product of their marginal density functions, which is $f(X_{i-1})f(X_i)$ [29]. Consequently, $H(X_i) - H(X_i|X_{i-1})$, which equals to the Kullback-Leibler divergence of $f(X_{i-1}, X_i)$ and $f(X_{i-1})f(X_i)$, is high. Based on Lemma 2, we conclude that: $T(\{X_1, \ldots, X_d\})$ is high. Lemma 2 also holds for any permutation of $\{X_i\}_{i=1}^{d}$. Hence, under any permutation of the dimensions of S, S has a high total correlation. This also means: The difference between the joint density function of S and the product of its marginal density functions is high w.r.t. the Kullback-Leibler divergence. Hence, subspaces fulfilling our notion likely are mutually correlated, not just pairwise correlated. Since many other correlation measures define mutual correlation based on the difference between the joint distribution and the product of marginal distributions [25], our subspaces are also likely mutually correlated under such correlation measures.

We now prove that our new notion of correlated subspaces is more general than that of APR:

Theorem 6: Let S be a subspace detected by APR using $Corr$ *(generalized for more than two dimensions) as correlation measure and given* $MAX\_NUM \leq K$*, then all of its pairs* $\{X_i, X_j\} \in \mathcal{S}_2$.

Proof: We use induction:
Let $S = \{X_1, \ldots, X_d\}$ be a subspace mined by APR.

Basis: When $d = 2$, it is clear that $S \in \mathcal{S}_2$.

Hypothesis: Assume Theorem 6 holds for $d = n \geq 2$.

Inference: We prove that Theorem 6 also holds for $d = n + 1$, i.e., we prove $\forall X_i \neq X_j \in S : \{X_i, X_j\} \in \mathcal{S}_2$. This is straightforward. For $X_i \neq X_j$, there exists an n-dimensional subspace $U \subset S$ such that $X_i, X_j \in U$ and U is included by APR in the output (cf., monotonicity property). Hence, $\{X_i, X_j\} \in \mathcal{S}_2$ according to the hypothesis. ■

Theorem 6 also holds for other correlation measures, e.g., the ones in [7], [15], [23], with $\mathcal{S}_2$ being formed according to the measure used. It implies that, given the same correlation measure and $MAX\_NUM \leq K$, all subspaces included in the final output of APR are also discovered by our mining scheme. This is because any two of their dimensions are pairwise correlated, i.e., they form cliques in the correlation graph. This shows that our mining scheme is more general than APR and, hence, can discover subspaces missed by APR. Note that a subspace satisfying the pairwise condition is not necessarily included in the final output of APR. However, the monotonicity restriction imposed by APR is only to reduce the runtime [23], and does not guarantee the quality of subspaces. Our empirical study also confirms this.

Having formally analyzed the theoretical properties of our notion of correlated subspaces, we now map the problem of mining subspaces in higher-level layers to maximal clique mining in the correlation graph. Consider an undirected correlation graph $\mathcal{G}_{\mathcal{D}}$ with nodes being the dimensions. An edge exists connecting two dimensions X_i and X_j iff $\{X_i, X_j\} \in \mathcal{S}_2$. A subspace of our interest then forms a clique in $\mathcal{G}_{\mathcal{D}}$. To avoid redundancy, we propose to mine only maximal cliques, i.e., subspaces are not completely contained in each other. We regard maximal cliques of $\mathcal{G}_{\mathcal{D}}$ as candidates for this step.

Given D dimensions, the worst-case complexity to find all maximal cliques is $O(3^{D/3})$. To ensure the practicality of 4S, we rely on a recent finding [4]. It states that the properties of a data set (e.g., distances between data points) are preserved after dimensionality reduction as long as the number of dimensions kept is $O(\log N)$. As a result, we set $K \leq D \log N$, i.e., $O(D \log N)$. Hence, the expected maximal degree of each node in $\mathcal{G}_{\mathcal{D}}$ is $O(\log N)$, i.e., each dimension can be part of subspaces (maximal cliques) with expected maximal dimensionality $O(\log N)$. Also, the expected degeneracy of $\mathcal{G}_{\mathcal{D}}$ is $O(\log N)$. Following [10], we obtain the following result:

Theorem 7: The expected time complexity of mining maximal cliques is $O(DN^{1/3} \log N)$. The expected number of maximal cliques is $O((D - \log N)N^{1/3})$.

Therefore, using our strategy, we can efficiently and directly mine high dimensional subspaces without knowledge loss. Further, we achieve this without traversing the subspace lattice in a levelwise manner. Note that our scheme is different from approaches imposing the maximal dimensionality of subspaces. This is because the maximal dimensionality is *implicitly* embedded in 4S, rather than explicitly. Further, 4S is not constrained by the $O(\log N)$ bound in practice. This is due to our MDL-based merge of subspaces, which reconstructs fragmented high dimensional correlated subspaces.

VI. Subspace Merge

We denote the set of dimensions, each of which belonging to at least one maximal clique, as $\{X_{r(j)}\}_{j=1}^{l}$. Also, $\{C_i\}_{i=1}^{m}$ is the set of maximal cliques. Due to the pairwise restriction of our subspace notion, subspaces (maximal cliques) obtained by mining $\mathcal{G}_{\mathcal{D}}$ may be projections of the same higher-dimensional correlated subspaces. To reconstruct such subspaces and to remove redundancy in the output, we merge subspaces into groups such that the new set of subspaces *guarantees completeness and minimizes redundancy*. To accomplish this, we first construct a binary matrix $\mathcal{B}$ with l rows and m columns. The rows are dimensions, and the columns are cliques. $\mathcal{B}_{ij} = 1$ iff X_i is in C_j, and 0 otherwise. We transform the subspace merge to grouping similar columns of $\mathcal{B}$, each final group constituting one subspace. We meet the merge requirements by applying the MDL-based algorithm in [19]. It compresses (groups) similar columns of $\mathcal{B}$ such that the total encoding cost of $\mathcal{B}$ given the grouping is minimal. We have:

Theorem 8: The subspace merge guarantees completeness and minimizes redundancy.

That is, our subspace merge ensures that its output subspaces capture all the subspaces produced by the second step (completeness). This stems from the fact that MDL guarantees a *lossless compression*. Thus, the original set of subspaces is compressed while ensuring that no information loss occurs. Besides, our algorithm heuristically selects the grouping of subspaces that minimizes the overall compression cost. For instance, if a grouping contains two very similar subspaces (i.e., redundant ones), our algorithm would not pick it since the merge of two subspaces can result in a better grouping with a lower encoding cost. Hence, redundancy is minimized.

According to [19], the total time complexity of this step is $O(lm^3)$, which is $O(l(D - \log N)^3 N)$. Nevertheless, the runtime in practice is much smaller because (a) the number of cliques is much smaller than the one stated in Theorem 7, (b) the number l of dimensions left is small compared to D, and (c) the subspace merge algorithm in [19] terminates early. Our experiments also point out that the runtime of this step is negligible compared to the first step. While APR can also apply this subspace merge, it does not achieve the same quality as 4S since it cannot reach high dimensional subspaces.

VII. Overall Complexity Analysis

The computation of $\mathcal{L}_2$ (using *Sketching*) costs $O(DN)$. The mining of $\mathcal{L}_k$ costs $O(DN^{1/3} \log N)$. The subspace merge costs $O(l(D - \log N)^3 N)$. Thus, the worst-case complexity of 4S is $O(l(D - \log N)^3 N)$. However, our experiments point out that the most time-consuming step is the computation of $\mathcal{L}_2$, which accounts for nearly 90% of the overall runtime. Hence, overall, we can say that 4S has $O(DN)$ average-case complexity. Our experiments also confirm that 4S has near-linear scalability with both N and D. This is a significant improvement from the $O(D^2 N + 2^k \cdot N \cdot MAX\_NUM)$ complexity of APR where k is the highest layer reached.

VIII. Experiments

We write 4S-M and 4S-S as 4S with *MultiPruning* and *Sketching*, respectively. We compare 4S with the following methods. FS as baseline in full-space. FB [18] using random subspaces for outlier mining. EC [7], CMI [23], and HICS [15] representing the APR-style methods. FEM [9] representing the unsupervised feature selection approaches. For all of these methods, we try to optimize their parameter settings. For our methods, we set $K = D \log N$. For 4S-S, we fix $s_1 = 10000$ and $s_2 = 2$. Setting s_2 to 2 follows the observation that smaller values for s_2 generally result in better accuracy [8].

We test the subspaces produced by all methods in: outlier detection with LOF [6], clustering with DBSCAN [11], and classification with the C4.5 decision tree. The first two areas are known to yield meaningful results when the subspaces selected have high correlations, i.e., include few or no irrelevant dimensions [7], [18], [21], [15], [23]. Hence, they are good choices for evaluating the quality of correlated subspaces. The third area is to show that correlated subspaces found by 4S are also useful for the supervised domain. For each method, LOF, DBSCAN, and C4.5 are applied on its detected subspaces and the results are combined, following [18] for LOF, [7] for DBSCAN, and [13] for C4.5.

We use synthetic data and 6 real-world data sets from the UCI Repository: the Gisette data about handwritten digits; HAR, PAMAP1, and PAMAP2 all sensor data sets with physical activity recordings; Mutant1 and Mutant2 containing biological data used for cancer prediction. Further, we use the facility management's database of our university (KIT) with energy indicators recorded from 2006 to 2011. More details are in Table I. Note that each of them has more than 1 trillion subspaces. This features a challenging search space w.r.t. dimensionality for all methods. Further, we assist future comparison, by providing data sets, parameters, and algorithms on our project website[1].

[1] http://www.ipd.kit.edu/~muellere/4S/

Data set	Size	Attributes	Classes
Gisette	13500	5000	2
HAR	10299	561	6
KIT	48501	540	2
Mutant1	16592	5408	2
Mutant2	31159	5408	2
PAMAP1	1686000	42	15
PAMAP2	1662216	51	18

TABLE I: Characteristics of real-world data sets. Each of them has more than 1 trillion subspaces.

A. Experiments on synthetic data

Quality on outlier detection. We have created 6 synthetic data sets of 10000 records and 100 to 1000 dimensions. Each data set contains subspace clusters with dimensionality varying from 8 to 24 and we embed 20 outliers deviating from these clusters. Our performance metric is the Area Under the ROC Curve (AUC), as in [18], [15], [16]. From Table II, one can see that 4S-M overall has the best AUC on all data sets. 4S-S in turn achieves the second-best performance. In fact, in most cases, 4S-M correctly discovers all embedded subspaces. Though 4S-S does not achieve that, its subspaces are close to the best ones, and it has better performance than other methods. We are better than FS, which focuses on the full-space where noisy dimensions likely hinder the detection of outliers. Our methods outperform FB, which highlights the utility of our correlated subspaces compared to random ones. Examining the subspaces found by APR-style methods (EC, CMI, and HICS), we see that they are either irrelevant, or they are low dimensional fragments of relevant subspaces. This explains their poor performance. FEM has low AUC since it only mines a single subspace and hence, misses other important correlated subspaces where outliers are present.

Quality on clustering. Synthetic data sets with 100 to 1000 dimensions are used again. Our performance metric is the F1 measure, as in [21], [20]. Table III displays clustering results of all methods. One can see that 4S-M and 4S-S have the best performance on all data sets tested. This again highlights the quality of subspaces found by our methods.

From the outlier detection and clustering experiments, we can see that 4S-S is a good approximation of 4S-M.

APR using subspace merge. For illustration, we only present the outlier detection and clustering results on the synthetic data set with 10000 records and 1000 dimensions. From Table IV, by applying the subspace merge, APR-style methods achieve better AUC and F1 values than without merge. Yet, our methods outperform all of them. This is because APR-style methods already face severe issue with reaching high dimensional subspaces. Thus, applying subspace merge in their case cannot bring much of improvement.

Scalability. Since FS and FB do not spend time for finding subspaces, we only analyze the runtime of the remaining methods. To test scalability w.r.t. dimensionality, we use data sets with 10000 data points and dimensionality of 100 to 1000. Based on Figure 2, 4S-S has the best scalability. FEM scales better than 4S-M because it only searches for a single subspace. Overall, 4S-S has near-linear scalability w.r.t. dimensionality, thanks to our efficient search scheme.

Task	4S-M	4S-S	EC	CMI	HICS
Outlier Mining (AUC)	**0.99**	0.92	0.76	0.49	0.44
Clustering (F1)	**0.91**	0.88	0.84	0.70	0.76

TABLE IV: Comparison with APR using subspace merge on the synthetic data set with 10000 records and 1000 dimensions. Highest values are in **bold**.

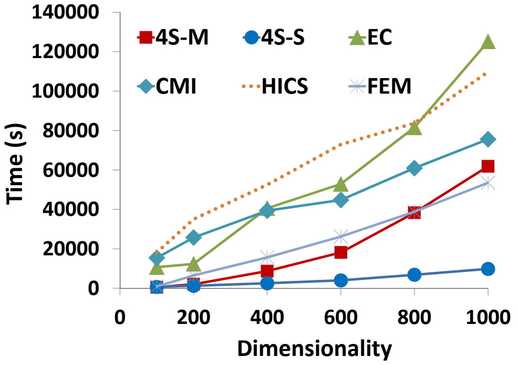

Fig. 2: Runtime w.r.t. dimensionality on synthetic data.

To test scalability to data size of all methods, we use data sets with 100 dimensions and sizes of 10000 to 100000. Figure 3 shows that 4S-S scales linearly and is more efficient than 4S-M. This is in line with our theoretical analysis.

We also note that the runtime of the first step in our methods dominates the other two steps. For example, on the data set of 10000 records and 1000 dimensions, 4S-S takes about 150 minutes for the first step and only 14 minutes for the remaining two steps.

From the results obtained, we can conclude that 4S-S achieves the efficiency goal while still ensuring high quality of subspaces found. From now onwards, we use 4S-S for the remaining experiments and write only 4S.

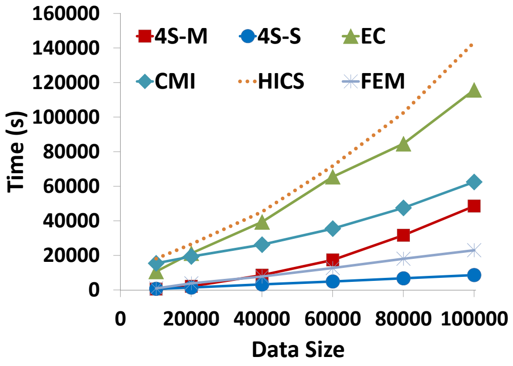

Fig. 3: Runtime w.r.t. data size on synthetic data.

Data set	4S-M	4S-S	FS	FB	EC	CMI	HICS	FEM
D100	**1.00**	**1.00**	**1.00**	0.65	0.90	0.46	0.43	0.50
D200	**1.00**	**1.00**	0.99	0.50	0.85	0.47	0.46	0.48
D400	**0.99**	0.98	0.96	0.51	0.83	0.46	0.45	0.63
D600	**0.99**	0.98	0.77	0.54	0.76	0.42	0.29	0.54
D800	**0.99**	0.87	0.75	0.61	0.74	0.43	0.40	0.59
D1000	**0.99**	0.92	0.81	0.47	0.75	0.46	0.40	0.64

TABLE II: AUC on outlier mining for synthetic data sets. Highest values are in **bold**.

Data set	4S-M	4S-S	FS	FB	EC	CMI	HICS	FEM
D100	**0.99**	**0.99**	0.72	0.95	0.67	0.50	0.80	0.76
D200	**0.89**	**0.89**	0.67	0.66	0.67	0.50	0.80	0.76
D400	**0.85**	0.83	0.67	0.81	0.67	0.80	0.77	0.75
D600	**0.96**	0.95	0.67	0.66	0.67	0.67	0.83	0.53
D800	**0.99**	0.93	0.67	0.67	0.67	0.67	0.83	0.74
D1000	**0.91**	0.88	0.67	0.67	0.83	0.67	0.74	0.75

TABLE III: F1 on clustering for synthetic data sets. Highest values are in **bold**.

Data set	4S	FS	FB	EC	CMI	HICS	FEM
Gisette	**0.77**	0.67	0.60	0.73	0.74	0.74	0.68
HAR	**0.67**	0.42	0.53	0.27	0.65	0.15	0.53
KIT	**0.73**	0.36	0.51	0.33	0.55	0.55	0.44
Mutant1	**0.62**	0.58	0.55	0.56	0.58	0.57	0.55
Mutant2	**0.64**	0.57	0.53	0.55	0.58	0.59	0.56
PAMAP1	**0.86**	0.54	0.47	*	*	*	0.48
PAMAP2	**0.87**	0.53	0.45	*	*	*	0.41

TABLE V: AUC on outlier mining for real-world data sets. Highest values are in **bold**. (*) means the result is unavailable due to excessive runtime.

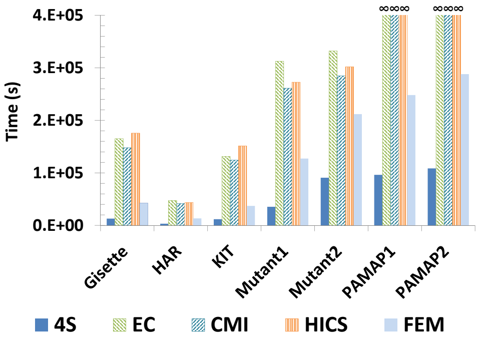

Fig. 4: Runtime (in seconds) of subspace search methods on real-world data sets. EC, CMI, and HICS did not finish within 5 days on the PAMAP data sets.

B. *Experiments on real data*

We apply all methods to two applications: outlier detection and classification. Clustering is skipped here since it conveys similar trends among the methods as with synthetic data.

Quality on outlier detection. As a standard procedure in outlier mining [18], [15], [16], the data sets used are converted to two-class ones, i.e., each contains only a class of normal objects and a class of outliers. This is done by either picking the smallest class or down-sampling one class to create the outlier class. The rest forms the normal class. From Table V, 4S achieves the best results. Its superior performance compared to other methods, including APR-style methods techniques (EC, CMI, and HICS), stems from the fact that 4S better discovers correlated subspaces where outliers are visible. For example, on the KIT data set, 4S finds subspaces where several consumption indicators of different buildings of the same type (e.g., office buildings, laboratories) cluster very well with a few exceptions, possibly caused by errors in smart-meter readings, or rare events (e.g., university holidays when energy consumption is low or large-scale physics experiments when electricity consumption is extremely high). These subspaces however are not discovered by other methods.

On the PAMAP1 and PAMAP2 data sets, we can only compare 4S against FS, FB, and FEM. This is because other methods take excessively long time without completing. These data sets contain data collected by sensors attached to human bodies when they perform different activities, e.g., walking, running, ascending stairs. The best AUC of 4S on both data sets once again implies that 4S successfully discovers high quality subspaces, which in turn assist in the detection of outliers. For example, the subspaces found by 4S on PAMAP1 exhibit correlations among the hand, chest, and ankle of human subjects. There are of course different grouping patterns representing different types of activities. In any case, such correlations let records representing transient activities become outliers. This is intuitive because those activities are very random and do not feature any specific correlation among different parts of human bodies [26].

In Figure 4 we show the wall-clock time (in seconds) for each subspace search method. Note that EC, CMI, and HICS did not finish within 5 days on the PAMAP data sets. The results show that 4S is much faster than all competitors.

Quality on classification. We here test 4S against the well-known Random Forest classifier [13], FEM for unsupervised feature selection, and CFS [12] for supervised feature selection. We skip other methods since previous experiments already show that 4S outperforms them. The classification accuracy (obtained by 10-fold cross validation) is in Table VI. Overall, 4S consistently yields better accuracy than Random

Data set	4S	Random Forest	FEM	CFS
Gisette	0.76	0.75	0.72	**0.84**
HAR	0.83	0.81	0.74	**0.85**
KIT	**0.97**	0.96	0.85	0.92
Mutant1	**0.99**	0.88	0.85	0.97
Mutant2	**0.99**	0.87	0.89	0.98
PAMAP1	**0.91**	0.71	0.69	0.87
PAMAP2	**0.93**	0.71	0.66	0.86

TABLE VI: Classification accuracy for real-world data sets. Highest values are in **bold**.

Forest and FEM. It is comparable to CFS which has access to the class label. The results obtained show that the correlated subspaces found by 4S are also useful for data classification.

IX. Conclusions

Mining high dimensional correlated subspaces is a very challenging but important task for knowledge discovery in multi-dimensional data. We have introduced 4S, a new scalable subspace search scheme that addresses the issue. In particular, through theoretical analysis, we have proven that the subspace search problem can be transformed to the problem of clique mining. This transformation not only avoids the high runtime complexity of Apriori methods but also leads to more general results (i.e., we detect correlated subspaces found by Apriori methods as well as subspaces missed by them). Both of these points have been proven in the paper.

Overall, compared to existing work, 4S embarks on a completely novel approach to efficiently solving the subspace search problem. Empirically, we have demonstrated that 4S scales to data sets of more than 1.5 million records and 5000 dimensions (i.e., more than 1 trillion subspaces). Not only being more efficient than existing methods, 4S also better detects high quality correlated subspaces that are useful for outlier mining, clustering, and classification.

Directions for future work include a systematic study of our search scheme with different correlation measures, and the integration of subspace merge into the correlation graph to perform an in-process removal of redundancy.

Acknowledgments

This work is supported by the German Research Foundation (DFG) within GRK 1194, by the Young Investigator Group program of KIT as part of the German Excellence Initiative, and by a Post-Doctoral Fellowship of the Research Foundation – Flanders (FWO).

References

[1] S. Achard, “Asymptotic properties of a dimension-robust quadratic dependence measure,” *Comptes Rendus Mathematique*, vol. 346, no. 3, pp. 213–216, 2008.

[2] C. Aggarwal and P. Yu, “Outlier detection for high dimensional data,” in *SIGMOD*, 2001, pp. 37–46.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic subspace clustering of high dimensional data for data mining applications,” in *SIGMOD*, 1998, pp. 94–105.

[4] N. Ailon and B. Chazelle, “Faster dimension reduction,” *Communications of the ACM*, vol. 53, no. 2, pp. 97–104, 2010.

[5] N. Alon, Y. Matias, and M. Szegedy, “The space complexity of approximating the frequency moments,” in *STOC*, 1996, pp. 20–29.

[6] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, “LOF: Identifying density-based local outliers,” in *SIGMOD*, 2000, pp. 93–104.

[7] C. H. Cheng, A. W.-C. Fu, and Y. Zhang, “Entropy-based subspace clustering for mining numerical data,” in *KDD*, 1999, pp. 84–93.

[8] A. Dobra, M. N. Garofalakis, J. Gehrke, and R. Rastogi, “Processing complex aggregate queries over data streams,” in *SIGMOD*, 2002, pp. 61–72.

[9] J. G. Dy and C. E. Brodley, “Feature selection for unsupervised learning,” *JMLR*, vol. 5, no. 8, pp. 909–921, 2004.

[10] D. Eppstein, M. Löffler, and D. Strash, “Listing all maximal cliques in sparse graphs in near-optimal time,” in *ISAAC (1)*, 2010, pp. 403–414.

[11] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *KDD*, 1996, pp. 226–231.

[12] M. A. Hall, “Correlation-based feature selection for discrete and numeric class machine learning,” in *ICML*, 2000, pp. 359–366.

[13] T. K. Ho, “The random subspace method for constructing decision forests,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[14] K. Kailing, H.-P. Kriegel, P. Kröger, and S. Wanka, “Ranking interesting subspaces for clustering high dimensional data,” in *PKDD*, 2003, pp. 241–252.

[15] F. Keller, E. Müller, and K. Böhm, “HiCS: High contrast subspaces for density-based outlier ranking,” in *ICDE*, 2012, pp. 1037–1048.

[16] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, “Outlier detection in arbitrarily oriented subspaces,” in *ICDM*, 2012, pp. 379–388.

[17] M. H. C. Law, M. A. T. Figueiredo, and A. K. Jain, “Simultaneous feature selection and clustering using mixture models,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 909–921, 2004.

[18] A. Lazarevic and V. Kumar, “Feature bagging for outlier detection,” in *KDD*, 2005, pp. 157–166.

[19] M. Mampaey and J. Vreeken, “Summarizing categorical data by clustering attributes,” *DMKD Journal*, vol. 26, no. 1, pp. 130–173, 2013.

[20] E. Müller, I. Assent, S. Günnemann, R. Krieger, and T. Seidl, “Relevant subspace clustering: Mining the most interesting non-redundant concepts in high dimensional data,” in *ICDM*, 2009, pp. 377–386.

[21] E. Müller, S. Günnemann, I. Assent, and T. Seidl, “Evaluating clustering in subspace projections of high dimensional data,” *PVLDB*, vol. 2, no. 1, pp. 1270–1281, 2009.

[22] E. Müller, M. Schiffer, and T. Seidl, “Statistical selection of relevant subspace projections for outlier ranking,” in *ICDE*, 2011, pp. 434–445.

[23] H. V. Nguyen, E. Müller, J. Vreeken, F. Keller, and K. Böhm, “CMI: An information-theoretic contrast measure for enhancing subspace cluster and outlier detection,” in *SDM*, 2013, pp. 198–206.

[24] M. Rao, Y. Chen, B. C. Vemuri, and F. Wang, “Cumulative residual entropy: A new measure of information,” *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1220–1228, 2004.

[25] M. Rao, S. Seth, J.-W. Xu, Y. Chen, H. Tagare, and J. C. Príncipe, “A test of independence based on a generalized correlation function,” *Signal Processing*, vol. 91, no. 1, pp. 15–27, 2011.

[26] A. Reiss and D. Stricker, “Towards global aerobic activity monitoring,” in *PETRA*, 2011.

[27] D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti, “Detecting novel associations in large data sets,” *Science*, vol. 334, no. 6062, pp. 1518–1524, 2011.

[28] F. Rusu and A. Dobra, “Sketches for size of join estimation,” *ACM Trans. Database Syst.*, vol. 33, no. 3, 2008.

[29] S. Seth, M. Rao, I. Park, and J. C. Príncipe, “A unified framework for quadratic measures of independence,” *IEEE Transactions on Signal Processing*, vol. 59, no. 8, pp. 3624–3635, 2011.

[30] A. J. Smola, A. Gretton, L. Song, and B. Schölkopf, “A hilbert space embedding for distributions,” in *ALT*, 2007, pp. 13–31.

[31] M. L. Yiu and N. Mamoulis, “Iterative projected clustering by subspace mining,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 176–189, 2005.

[32] L. Yu and H. Liu, “Efficient feature selection via analysis of relevance and redundancy,” *Journal of Machine Learning Research*, vol. 5, pp. 1205–1224, 2004.