Subspace Clustering Meets Dense Subgraph Mining: A Synthesis of Two Paradigms

Stephan Günnemann  Ines Färber  Brigitte Boden  Thomas Seidl

RWTH Aachen University, Germany
{guennemann, faerber, boden, seidl}@cs.rwth-aachen.de

Abstract—Today’s applications deal with multiple types of information: graph data to represent the relations between objects and attribute data to characterize single objects. Analyzing both data sources simultaneously can increase the quality of mining methods. Recently, combined clustering approaches were introduced, which detect densely connected node sets within one large graph that also show high similarity according to all of their attribute values. However, for attribute data it is known that this full-space clustering often leads to poor clustering results. Thus, subspace clustering was introduced to identify locally relevant subsets of attributes for each cluster.

In this work, we propose a method for finding homogeneous groups by joining the paradigms of subspace clustering and dense subgraph mining, i.e. we determine sets of nodes that show high similarity in subsets of their dimensions and that are as well densely connected within the given graph. Our twofold clusters are optimized according to their density, size, and number of relevant dimensions. Our developed redundancy model confines the clustering to a manageable size of only the most interesting clusters. We introduce the algorithm GAMER for the efficient calculation of our clustering. In thorough experiments on synthetic and real world data we show that GAMER achieves low runtimes and high clustering qualities.

I. INTRODUCTION

In the knowledge discovery process, clustering is an established technique for grouping similar objects while separating dissimilar ones. In traditional clustering the similarity of objects corresponds to the similarity of attribute values. For example, sensors in sensor networks are grouped according to similar measurements as temperature and humidity. Another type of information is graph data, for which graph clustering methods determine groups of nodes that correspond to densely connected subgraphs within a single graph. For example, functional related genes are grouped together in gene interaction networks. Here the similarity of objects corresponds to the density of connected subsets. Consequently, two different types of information can be used to determine clusters in the data.

In many applications information of both categories is available for the objects, graph information as well as attribute data. Both together can be modeled as a vertex-labeled graph such that vertices represent objects, edges represent relations between them, and feature vectors associated to the vertices represent the attributes for each object (cf. Fig. 1). Since the separate use of both data types can lead to highly differing clusterings and thus contradictory results, their simultaneous use for the process of clustering promises more meaningful and accurate results. Therefore, combined clustering approaches have been recently introduced, which try to determine groups that are densely connected within the graph as well as similar according to their attribute values.

The main problem of almost all combined clustering approaches, however, is the consideration of all attribute dimensions for determining the similarity. Often, some dimensions are not relevant for all clusters, which is why clusters are located in subsets of the dimensions. E.g. in social networks, it is very unlikely that people are similar within all of their characteristics. A continuative aspect is the curse of dimensionality [1] leading to questionable clustering results in the full-space. As a solution subspace clustering methods were introduced, especially useful for high dimensional data like genes. Consistently, also combined clustering models should analyze subsets of the attributes.

Our novel approach combines graph data and attribute data to identify groups according to their density of connections as well as similarity of attribute values. In contrast to other approaches, however, we consider subsets of the dimensions to realize meaningful similarity determination. A clustering procedure like this is advantageous for a variety of applications: Highly connected groups of people in social networks (graph density) can be used for target and viral marketing based on their specific preferences (attribute subset). In sensor networks, an aggregated transmission of specific sensor measurements (attribute subset) of communicating sensors (graph density) leads to improved energy efficiency and thus longer lifetime of the network.

A sound combination of the paradigms subspace clustering and dense subgraph mining has to be unbiased in the sense that none of the paradigms is preferred over the other. However, most combined clustering models prioritize one of them. In Fig. 1 for example the largest clique is \{2, 3, 5, 6\}; unfortunately, these nodes show similar behavior
only in one of their attributes. Even worse, preferring just high dimensional clusters leads to \{1, 4, 6\}; this cluster cannot be reconciled with the graph structure. Obviously the cluster properties ‘density’, ‘dimensionality’, and ‘size’ are usually contradictory and a clustering model has to realize a reasonable trade-off. The challenge tackled by our approach is the optimization of all three goals simultaneously to ensure their equality. This enables both paradigms to act on an equal footing in order to obtain meaningful and consistent clusters. Node group \{1, 2, 5, 6\} and node group \{2, 3, 5\} could be possible clusters for such an optimization. In both clusters all nodes have similar values in 2 attributes and the density of the subgraphs is negligibly smaller than in cliques.

A further observation is that overlaps between clusters are quite reasonable. While the cluster \{1, 2, 5, 6\} might be of interest for video game producers, the cluster \{2, 3, 5\} might be of interest for sports wear retailers. Similarly, genes can belong to more than one functional module and sensors to more than one aggregation unit. Highly overlapping clusters, however, often imply similar interpretations and thus a more than one aggregation unit. Highly overlapping clusters, where for each cluster exponentially many subspace projections exist, avoiding redundancy is impossible. In a clustering model, where for each cluster exponentially many subspace projections exist, avoiding redundancy is indispensable. Therefore, treating redundancy is also needed for the combination of subspace clustering and subgraph mining. Our model successfully avoids redundancy in the clustering result, while generally allows the clusters to overlap.

The main contributions of our work are the following:

- A novel cluster model, equitably joining the paradigms of subspace clustering and dense subgraph mining.
- A novel clustering model, including a redundancy model to avoid unnecessary increase of the result set while permitting overlaps between clusters in general.
- The algorithm GAMER, exploiting various pruning strategies for the efficient calculation of the defined clustering.

II. RELATED WORK

Traditional clustering for vector data evaluates clusters w.r.t. all attributes; they do not scale to high dimensional data due to the irrelevance of some attributes [1]. Thus, subspace clustering methods detect relevant subspace projections for each cluster individually [2]. None of the proposed subspace clustering methods, however, considers graph data.

Clustering graph data has been done in different ways [3]. We focus on mining densely connected subgraphs in one large graph. Unlike to partitioning methods others assume that the graph naturally divides into (possibly overlapping) subgraphs of certain patterns as γ-quasi-cliques [4], [5]. None of these approaches, however, considers attribute data.

A few methods consider both information types. Some [6], [7] consider attributes only in a post-processing step. [8] transforms the graph and the attributes to a single, combined distance function. The resulting clusters have no specific graph structure. Contrarily, [9] transforms the attribute data into a graph. Both methods [8], [9] cannot detect similarities based on subsets of attributes and they determine almost disjoint clusters. In [10] attribute values are modeled as additional nodes. Although objects in one cluster do not necessarily correspond in all attributes, no specific relevant attribute subset can be defined for the clusters. CoPaM [11] is the only approach so far that deals with subspace clustering and dense subgraph mining. Though, it considers the density and the subspace cardinality only as minimal thresholds and solely optimizes the number of nodes.

One further drawback of all combined clusterings methods is their missing or limited redundancy handling. In our novel approach, redundant clusters are removed from the output.

III. A COMBINED CLUSTERING MODEL

This section presents our model for detecting densely connected subgraphs that exhibit feature similarity in subsets of the dimensions – called twofold clusters. To this end, we model attribute data together with graph data by using a vertex-labeled graph \( G = (V, E, l) \) with vertices \( V \), edges \( E \subseteq V \times V \) and a labeling function \( l : V \rightarrow \mathbb{R}^d \) where \( \text{Dim} = \{1, \ldots, d\} \) is the set of dimensions. We assume an undirected graph without self-loops. We use \( l(O) = \{l(o) \mid o \in O\} \) to denote the set of vectors associated to the set of vertices \( O \subseteq V \) and \( x[i] \) to refer to the \( i \)-th component of a vector \( x \in \mathbb{R}^d \). Based on this input graph, we introduce in Sec. III-A the definition of single clusters and in Sec. III-B the overall clustering to avoid redundancy.

A. Cluster definition

Our twofold clusters should simultaneously represent subspace clusters and dense subgraphs. Therefore we combine established definitions of both paradigms. For subspace clustering we choose the effective and efficient cell-based methods [12]. A subspace cluster is a set of objects with a set of relevant dimensions. Within the relevant dimensions the objects have to be very similar, i.e. the variation of their attribute values is restricted to a maximal width \( w \).

**Definition 1:** Subspace cluster property

Given a set of vectors \( X \subseteq \mathbb{R}^d \) and a set of dimensions \( S \subseteq \text{Dim} \), the tuple \( (X, S) \) is a subspace cluster if

\[
\forall i \in S : \forall x_1, x_2 \in X : |x_1[i] - x_2[i]| \leq w
\]

\[
\forall i \in \text{Dim} \setminus S : \exists x_1, x_2 \in X : |x_1[i] - x_2[i]| > w
\]

In Fig. 2 the vectors \( l(O_1) \) and the dimensions \( S_1 \) as well as \( (l(O_4), S_4) \) are valid subspace clusters for \( w = 0.5 \). For identifying dense subgraphs we use the notion of quasi-cliques [4]. Within a quasi-clique \( O \) each vertex \( v \in O \) is connected to a certain minimal percentage of vertices of \( O \).

**Definition 2:** Quasi-clique property

A set of vertices \( O \subseteq V \) within a graph \( G = (V, E, l) \) is a \( \gamma \)-quasi-clique if \( \min_{v \in O} \{ \text{deg}^O(v) \} \geq \gamma \cdot |O| - 1 \) where \( \text{deg}^O(v) \) is the degree of vertex \( v \) within the set \( O \), i.e. \( \text{deg}^O(v) = |\{o \in O \mid (v, o) \in E\}| \). The density of a quasi-clique is given by \( \gamma (O) = \min_{v \in O} \{ \text{deg}^O(v) \} / (|O| - 1) \).
In Fig. 2 the set $O_2$ is the largest 0.5-quasi-clique within the graph. Quasi-cliques describe object sets based on their connectivity; potentially they have only few (or even no) relevant dimensions. Contrarily, subspace clusters describe object sets based on their similarity in subspaces; the underlying subgraph is potentially not dense (or even not connected). Thus, our twofold clusters have to fulfill both properties simultaneously.

**Definition 3:** Twofold cluster
A twofold cluster $C = (O, S)$ is a set of vertices $O \subseteq V$ and a set of dimensions $S \subseteq Dim$ with the following properties:
- $(l(O), S)$ is a subspace cluster with $|S| \geq s_{\text{min}}$
- $O$ fulfills the quasi-clique property with $\gamma(O) \geq \gamma_{\text{min}}$
- the induced subgraph of $O$ is connected and $|O| \geq n_{\text{min}}$

Since for $\gamma_{\text{min}} < 0.5$ a quasi-clique needs not to be connected [5], we enforce the connectivity to ensure reasonable clustering structures. In Fig. 2 we get the twofold cluster $(O_3, S_3)$ by $n_{\text{min}} = 3, s_{\text{min}} = 2, \gamma_{\text{min}} = 0.5$. The previous examples for subspace clusters or quasi-cliques, however, are not valid ones because at least one of the properties is violated. Def. 3 yields more meaningful clusters.

**B. Clustering definition**

With the introduced definition we are able to determine the set of all valid twofold clusters $\text{Clusters}$. Without any constraints this set can be large because we permit overlapping clusters in general. For example, by choosing $s_{\text{min}} = 1$ the cluster $(O_2, S_2)$ in Fig. 2 is also valid. This cluster, however, intuitively provides only little novel information compared to the cluster $(O_3, S_3)$; the vertices differ only marginally and we have less dimensions. By introducing a clustering definition, i.e. by determining a meaningful subset $\text{Result} \subseteq \text{Clusters}$, we focus on the most interesting clusters. Redundant clusters, providing only little additional information, are not included. The interestingness of clusters is presented in Sec. III-B1, our redundancy model in Sec. III-B2, and the overall clustering $\text{Result}$ in Sec. III-B3.

1) **Quality function:** The interestingness of our twofold clusters cannot be solved trivially. Usually, subspace clustering models try to maximize the dimensionality of clusters while dense subgraph methods maximize either the number of vertices or the density of subgraphs. Optimizing all these properties, however, results in contrary objective functions. For example it is possible that a set of vertices has a high density and by adding just one vertex to this set the density dramatically drops to a low value. It is thus mandatory to trade off these characteristics of clusters to realize a sound and unbiased synthesis of subspace and subgraph mining. Our quality function rates the interestingness of a twofold cluster based on these three aspects.

**Definition 4:** Quality of a twofold cluster
Given a twofold cluster $C = (O, S)$, the quality of $C$ is defined by

$$Q(C) = \gamma(O)^a \cdot |O|^b \cdot |S|^c$$

By this quality function we get a flexible model. With $a = b = c = 1$ for example we rate all aspects equally. In Fig. 2 we thus get for $C_2 = (O_2, S_2)$ and $C_3 = (O_3, S_3)$ the quality values $Q(C_2) = 0.5 \cdot 7 \cdot 1 = 3.5$ and $Q(C_3) = 10.8$.

2) **Redundancy model:** Our redundancy model identifies clusters that provide no or only little additional information. Previous methods use the maximality of patterns (w.r.t. objects or dimensions) to exclude patterns corresponding to subsets of the objects/dimensions. This, however, is not meaningful. First, the maximal clusters can differ only in few objects/dimensions; in this case they provide no novel knowledge and the result size can still be large. Second, the maximal clusters are not necessarily the most interesting clusters. Consequently, the quality function has to be involved to eliminate redundancy. A cluster $C$ can only be redundant compared to a cluster $\overline{C}$ if $Q(C) < Q(\overline{C})$. If the cluster $C$ had a higher quality, then it should not be reported as redundant w.r.t. $\overline{C}$; the user is more interested in $C$.

Furthermore, the cluster $C$ should not describe novel structural information. In our context, the objects as well as the relevant dimensions of $C = (O, S)$ should already be covered by most parts by the cluster $\overline{C} = (\overline{O}, \overline{S})$. If the fraction $\frac{|\overline{O} \cap S|}{|\overline{O}|}$ is large, only a small percentage of $C$’s objects are not contained in $\overline{C}$; we do not have a large information gain based on the object grouping of $C$. The same holds for the set of relevant dimensions. If all three indicators are valid, the cluster $C$ is redundant w.r.t. $\overline{C}$. We denote this by $C \prec_{\text{red}} \overline{C}$ and we formally define:

**Definition 5:** Binary redundancy relation
Given the redundancy parameters $r_{\text{obj}}, r_{\text{dim}} \in [0, 1]$, the binary redundancy relation $\prec_{\text{red}}$ is defined by:

For all twofold clusters $C = (O, S), \overline{C} = (\overline{O}, \overline{S})$:

$C \prec_{\text{red}} \overline{C} \Leftrightarrow Q(C) < Q(\overline{C}) \wedge \frac{|\overline{O} \cap S|}{|\overline{O}|} \geq r_{\text{obj}} \wedge \frac{|S|}{|S|} \geq r_{\text{dim}}$

The higher the redundancy parameter $r_{\text{obj}}, r_{\text{dim}}$, the more objects/dimensions of $C$ have to be covered by $\overline{C}$. For the extremal case of $r_{\text{obj}} = r_{\text{dim}} = 1$, $C$’s objects/dimensions have to be a subset of the ones of $\overline{C}$. In this case only few clusters are redundant. By choosing smaller values, the redundancy occurs more often. Considering $C_2 = (O_2, S_2)$
and $C_3 = \{O_3, S_3\}$ in Fig. 2 we get $C_2 \prec_{\text{red}} C_3$ for $r_{obj} = r_{dim} = 0.5$. By choosing $r_{obj} = 1$, $r_{dim} = 0.5$, however, none of the clusters is redundant to the other.

3) Overall clustering: After defining a binary relation for pairwise redundancy of clusters, we now define the overall clustering, i.e. given all twofold clusters $\text{Clusters}$ we want to get a meaningful subset $\text{Result} \subseteq \text{Clusters}$.

Since redundant clusters provide only little novel information, they are not beneficial for the user. Thus, the final clustering has to be redundancy-free. Note that our binary redundancy relation is non-transitive, i.e. we can have clusters $\{C_a, C_b, C_c\}$ with $C_a \prec_{\text{red}} C_b$, $C_b \prec_{\text{red}} C_c$ but not $C_a \prec_{\text{red}} C_c$. Thus, simply removing clusters that are redundant to at least one other cluster is too naive, e.g. in the previous example we would get $\{C_c\}$. However, the result $\{C_a, C_c\}$ is more meaningful because these clusters are also pairwise non-redundant and we get the additional cluster $C_a$.

Evidently, we need our result to fulfill a second property – the maximality property: For each cluster $C$ not selected for the result set, there is at least one selected cluster to which $C$ is redundant. Thus, if we selected $C$ the redundancy-free property would be violated. Our overall clustering result is:

Definition 6: Optimal twofold clustering

Given the set of all twofold clusters $\text{Clusters}$, the optimal twofold clustering $\text{Result} \subseteq \text{Clusters}$ fulfills

- $\neg \exists C_i, C_j \in \text{Result} : C_i \prec_{\text{red}} C_j$ (redundancy-free)
- $\forall C_i \in \text{Clusters} \setminus \text{Result} : \exists C_j \in \text{Result} : C_i \prec_{\text{red}} C_j$ (maximality)

In our previous example $\{C_a, C_b, C_c\}$ we get the desired $\text{Result} = \{C_a, C_c\}$. Our optimal twofold clustering confines the output to the most interesting, non-redundant clusters.

IV. THE GAMER ALGORITHM

In the following we give a short overview of our algorithm GAMER (Graph & Attribute Miner), which determines the optimal twofold clustering according to Def. 6. We develop pruning methods based on clustering and clustering properties.

Pruning based on cluster definition. Naively, we would have to check $2^{|V|}$ many subsets $O \subseteq V$ whether they fulfill our cluster definition. We use Def. 3 systematically for early pruning of vertex sets that cannot lead to valid clusters. Initially, we exclude vertices and edges that cannot belong to any valid cluster according to their vertex degree or subspace similarity of adjacent nodes. By this initial pruning the graph gets more sparse and can be more efficiently analyzed.

To enumerate the remaining vertex sets in the graph we use the set enumeration tree [13]. A tree for a graph with four vertices is depicted in Fig. 3. Each node of the tree represents a set of vertices $O \subseteq V$ and is associated with a candidate set $\text{cand}_O$. A child node $O'$ extends its parent node $O$ through one of the vertices in $\text{cand}_O$. Thus, the subtree of a node $O$ represents all potential clusters $O \subseteq O \cup \text{cand}_O$. By pruning the tree we narrow down the search space. Note that the quasi-clique property is not monotone and we cannot simply prune a subtree if the parent node is not a valid cluster. Instead we prune a vertex $v$ from the candidate set of a node $O$, if $\{v\} \cup O$ could never result in a valid cluster, not even by adding further vertices. If we were able to remove e.g. the vertex $v_3$ from the set $\text{cand}_{\{v_1\}}$, the highlighted subsets in Fig. 3 would disqualify themselves as clusters without further analysis. For pruning, we combine the information about the already identified subspaces, the vertex degree, and the maximal diameter of quasi-cliques.

Pruning based on clustering definition. Apart from the cluster definition, which is used to avoid analyzing invalid clusters, we also use the clustering definition for pruning subtrees: we avoid generating valid clusters that, however, are redundant and thus are not allowed for the result. Thus, we identify whole subtrees of the set enumeration tree that will only contain redundant clusters w.r.t. a reference cluster $C$ (cf. Fig. 3). However, since our redundancy-relation is non-transitive, we cannot just discard these subtrees. Instead, we store them as so-called cluster collections, representing the set of clusters $\text{Coll}$ of the corresponding subtree. Thus, if $C$ is included in the final clustering later on, we can discard $\text{Coll}$, otherwise we have to traverse the subtree. This pruning technique enables an efficient generation of the overall result by avoiding to enumerate redundant clusters.

Overall algorithm. In GAMER the set enumeration tree is nested with a priority queue, which ranks the clusters by their quality. We start a depth-first traversal of the set enumeration tree at the root and insert the generated clusters and cluster collections into the queue. Afterwards, we process the queue as illustrated in Fig. 4. At each time we select the top ranked object out of the queue and we check the redundancy w.r.t. clusters in the actual result set $\text{Result}$. If it is redundant we discard the object. Otherwise, if it is a cluster, we add it to $\text{Result}$. If it is a collection, we traverse the set enumeration tree at the subtree represented by the collection. We refine this subtree, i.e. we generate further clusters and collections that are inserted into the queue.
This algorithmic processing guarantees to determine the optimal twofold clustering according to Def. 6. Due to space limitations we omit the proof. Overall, we use the subspace property, the quasi-clique property and the redundancy model simultaneously to achieve an efficient execution.

V. EXPERIMENTS

Setup. We compare GAMER against CoPaM [11] and an adaption of Cocain [5]. Originally, Cocain considers a set of graphs to find quasi-cliques that several graphs have in common. Since we use attribute data, we generate one graph per dimension and retain only those edges of the original graph connecting nodes with similar attribute values in this dimension. Thus, our Cocain° method simulates subspace clustering. Furthermore, we implement two baseline algorithms to analyze the efficiency of GAMER. These baseline algorithms generate the same result as GAMER but do not simultaneously use the subspace and subgraph properties for generation and pruning of clusters. SeqSubGraph starts by generating all quasi-cliques and after this it checks the subspace property. SeqSubSpace generates all subspace clusters and checks afterwards the quasi-clique property.

We use public available real world data including genes, patent data, the Arxiv database, and the DBLP database. Furthermore, we use synthetic data, by default with 80 clusters, each with 15 nodes, a density of 0.6 and 5-10 relevant dimensions out of 20 dimensions. 6% of the clusters nodes overlap. We provide all datasets and their descriptions as well as executables and parameter settings on our website¹. Efficiency is measured by runtime (Opteron 2.3GHz CPUs, Java 64 bit) and clustering quality by the F1 value [12].

Scalability on synthetic data. In Fig. 5 we increase the database size by varying the number of clusters. GAMER is several orders of magnitude faster than the competing approaches (please note the logarithmic scale). Especially, SeqSubSpace increases heavily and is not applicable on data sets with more than 1500 nodes. This approach generates a huge amount of subspace clusters, that, however, are not connected and hence are not valid. By incorporating the graph structure GAMER can early reject those node sets.

Fig. 6 analyzes the effects when the dimensionality of the data is increased. The slopes of all curves are nearly identical. The high runtime of SeqSubSpace indicates that graph based pruning is more effective than subspace pruning. However, a combination of both is even better because the absolute runtime of GAMER is still the lowest.

Quality on synthetic data. In the following we exclude the baseline methods since the results are identical to GAMER. Fig. 7 analyzes the robustness w.r.t. noise. We add noise objects that do not belong to any cluster (25% w.r.t. the former objects) and edges that connect different clusters. GAMER is nearly not influenced by noise and gets high quality. The qualities of CoPaM and Cocain° decrease. By adding noise, supersets of the hidden clusters become dense. Since both maximize the number of nodes, they misleadingly detect these supersets and include noise. GAMER, however, identifies the correct clusters since they have higher density and more relevant dimensions. Our trade-off between the characteristics leads to better quality. If GAMER was parameterized to prefer maximal clusters, it could also obtain perfect qualities in settings with few noise; however, by trading off the aspects we get high qualities for all settings.

In Fig. 8 we increase the degree of overlap, i.e. the maximal number of clusters that a node can belong to (for degree 1, clusters do not overlap). GAMER can handle high overlap by focusing on the non-redundant clusters. CoPaM and Cocain° fail for these settings; the quality drops or the algorithms are not applicable due to extreme memory usage. Node sets combined of different clusters are wrongly identified as clusters and are not rejected as redundant. Our redundancy model prevents to generate these clusters.

In Fig. 9 we vary the minimal density $\gamma_{min}$ used in all methods. We generate clusters with densities between 0.5 and 0.8. For $\gamma_{min}=1$ none of the algorithms gets perfect quality. Since the hidden clusters have lower densities they cannot be detected completely. If the minimal density is decreased, the quality of GAMER increases. We detect more and more hidden clusters. For a sufficiently small $\gamma_{min}$ the quality remains high. Although several further sets fulfill this minimal density, our redundancy model focuses on the true ones. Cocain° shows a different behavior: it dramatically drops for low density values. For a low threshold many clusters are regarded as dense and these redundant clusters are not excluded. CoPaM shows poor quality based on similar reasons. Due to our redundancy model, GAMER is more robust w.r.t. $\gamma_{min}$. A further drawback of CoPaM

¹http://www.dme.rwth-aachen.de/Gamer
and Cocain° is, that γ_{min} has to be larger than 1/3 or 1/2. GAMER can operate with arbitrary densities. All experiments indicate that GAMER gets high clustering qualities by confining the result to the most interesting clusters.

**Quality on real world data.** For real world data a ground truth is usually not given. For the gene data, however, we can use biological categories determined by Go-Miner [14] as the hidden clusters (also done in [11]). For this experiment GAMER obtains the highest quality results (F1: 0.43). The limited models of CoPaM (0.21) and Cocain° (0.19) are not able to detect meaningful clusters. Furthermore, we calculate the results of methods considering only one paradigm, i.e., subgraph mining (maximal quasi-cliques, Quick [4]) or subspace clustering (Proclus [15]). Their low qualities (0.25, 0.06) indicate that a synthesis of both paradigms – as our model does – can effectively increase the clustering quality.

For our remaining real world data no hidden clusters are given. Thus, we analyze in Fig. 10 different properties of the clustering results determined by GAMER, CoPaM and Cocain° (not listed methods did not finish within 2 days). First, we see that the runtimes of CoPaM and Cocain° are orders of magnitude higher than the runtime of GAMER. Considering the number of generated clusters, the huge result size of CoPaM becomes apparent. CoPaM excludes nearly no clusters; the redundancy model is too simple and clusters highly overlap. GAMER permits an overlap of clusters only if it does not entail redundancy. Although our approach implements a trade-off between different criteria while CoPaM concentrates on maximizing the number of nodes per cluster, GAMER determines clusters of comparable size but considerably higher density. Especially for the gene data, we see that the dimensionality is higher too.

**VI. Conclusion**

We introduce the method GAMER for finding homogeneous groups of objects regarding combined graph and attribute data. Our twofold clusters join the advantages of subspace clustering and dense subgraph mining. We simultaneously account for the density, the size and the number of relevant dimensions of each cluster to obtain the most interesting ones. Our redundancy model confines the clustering by excluding clusters that provide no additional information. Overall, we include only the most interesting and non-redundant clusters. Thorough experiments demonstrate that GAMER constantly outperforms the competing approaches in terms of efficiency and clustering quality.

**Acknowledgment.** This work has been supported by the UMIC Research Centre, RWTH Aachen University, Germany.

**References**


