Finding Succinct Representations For Clusters

Aparna Gupta

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science & Applications

Madhav V. Marathe, Chair
Anil S. Vullikanti
Samarth Swarup

May 10, 2019
Blacksburg, Virginia

Keywords: Explainable AI, Interpretable AI, Clustering, Integer Programming, Rounding Algorithm

Finding Succinct Representations For Clusters

Aparna Gupta

(ABSTRACT)

Improving the explainability of results from machine learning methods has become an important research goal. In this thesis, we studied the problem of making clusters more interpretable using a recent approach by Davidson et al., and Sambaturu et al., based on succinct representations of clusters. Given a set of objects S, a partition π of S (into clusters), and a universe T of descriptors such that each element in S is associated with a subset of descriptors, the goal is to find a representative set of descriptors for each cluster such that those sets are pairwise-disjoint and the total size of all the representatives is at most a given budget. Since this problem is NP-hard in general, Sambaturu et al. developed a suite of approximation algorithms for the problem. In this thesis we have done empirical analysis of the approximation algorithms developed by Sambaturu et al., and implemented various rounding schemes for a comparative study. We also show applications to explain clusters of genomic sequences that represent different threat levels

Finding Succinct Representations For Clusters

Aparna Gupta

(GENERAL AUDIENCE ABSTRACT)

Improving the explainability of results from machine learning methods has become an important research goal. Clustering is a commonly used Machine Learning technique which is performed on a variety of datasets. In this thesis, we have studied the problem of making clusters more interpretable; and have tried to answer whether it is possible to explain clusters using a set of attributes which were not used while generating these clusters.

Dedication

I dedicate this thesis to my parents and my sister, who has always been a wonderful source of happiness in my life.

Acknowledgments

I would first like to express my gratitude and thank my thesis advisor Dr Anil Vullikanti, for his untiring support and valuable guidance. He consistently allowed this thesis to be my work but also steered me in the right direction when needed. I also would like to thank my other advisors Dr Madhav Marathe & Dr Samarth Swarup, and all colleagues at Biocomplexity Institute of Virginia Tech who provided vital insights and expertise that greatly assisted this research.

Furthermore, I would like to further acknowledge Dr S. S. Ravi, Dr Andrew Warren and Prathyush Sambaturu for their enthusiastic participation and helpful comments on this thesis.

Finally, I express my profound gratitude to my family and loved ones who provided me with unfailing support and spiritual encouragement through the process of researching, editing this thesis, and indeed throughout my journey at Virginia Tech. This achievement would not have been possible without them. Thank You.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
LP	Linear Programming
ML	Machine Learning

Chapter 1

Introduction

1.1 Motivation

As AI and machine learning (ML) methods become pervasive across all domains from healthcare to urban planning, there is an increasing need to make the results of such methods more interpretable. Providing such explanations has now become a legal requirement in some countries [GF16]. Many researchers are investigating this topic using the framework of supervised learning, particularly for methods in deep learning (see e.g., [Pro17, Pro18]). Clustering is a commonly used unsupervised ML technique (see e.g., [BGLL08, Bol13, For10, Tan18, HPK11, ZMJM14]) and is routinely performed on diverse kind of datasets to identify the inherent grouping in unlabeled data. In these techniques, the data points are classified into multiple groups based on a similarity score. Since the data is unlabeled, there is no specific criteria for a good clustering technique. It depends solely on the user as to what criteria they use based on their dataset and needs. This can often make clusters hard to interpret, especially in a post-hoc analysis. For instance, given a set of points, with a notion of distance between them, the objective is to group these points into some group of clusters so that, members of a cluster are close/similar to each other and members of different clusters are dissimilar. A clustering algorithm will assign these points into clusters based on a distance measure (Euclidean, Cosine, Jaccard, etc.) rather than the associated attributes.

Is distance the only measure to group these points into similar groups? What if additional attributes are available for each data point? Can they be used as an additional source of information? A question that follows is whether it is possible to *explain* a given set of clusters, using additional attributes which, crucially, were not used in the clustering procedure.

A motivation for our work is to understand the *threat levels* of pathogens for which genomic sequences are available in [JGK18, JK18, HF18, RW18]. Researchers have been able to identify some genomic sequences as coming from harmful pathogens through lab experiments and bioinformatics analysis. Understanding what attributes make some sequences harmful, and distinguishing them from harmless sequences corresponds to the problem of interpreting the clusters.

Once the clustering is done, supervised machine learning techniques can be used to interpret the results of these clusters. One way is to convert the problem into a feature selection problem. In this case, the question can be "What is the best way to describe a cluster?" or "What distinguishes a cluster from all other clusters?" Another way is to apply Logistic Regression to understand the structure of data and predict class labels. We analysed the results of a Multinomial Logistic Regression model and observed that for a threat dataset with "248 sequences" and "4636" attributes, "59" attributes were used to describe cluster 1 (with 75 sequences) and "70" attributes were used to describe cluster 2 (with 175 sequences). Although Logistic Regression provided a reasonable set of descriptors to describe each cluster, the attribute set for each cluster is quite large. Logistic Regression assigns weights (called coefficients) to each feature. The higher the weight, the more important a feature is. However, there can be situations (for instance in multinomial regression) where weights associated with a feature are almost similar for each class label. A natural question that arises here is which label does a particular feature explains better? This is when the process of associating features with a class label becomes arduous. Also, the weights associated with a feature range from $-\infty$ to ∞. How large should a feature weight be, to interpret its importance? Our objective is to overcome this shortcoming by selecting important attributes such that: a minimum number of attributes selected provides high coverage and the disjointness condition is met. Therefore, our objective is to select a minimum number of attributes such that we have high coverage and the disjointness condition is met.

To achieve the above objectives, this thesis focuses on the formulation proposed by Davidson et al. [DGR18], in which they have presented the following formulation of the *Cluster Description Problem* for explaining a given set of clusters. Let $S = \{s_1, \ldots, s_n\}$ be a set of n objects. Let $\pi = \{C_1, \ldots, C_k\}$ be a partition of S into $k \geq 2$ clusters. Let T be the universe of tags such that each object $s_i \in S$ is associated with a subset $t_i \subseteq T$ of tags. A **descriptor** X_ℓ for a cluster C_ℓ $(1 \leq \ell \leq k)$ is a subset of T. An object s_i in cluster C_ℓ is said to be *covered* by the descriptor X_ℓ if at least one of the tags associated with s_i is in D_ℓ. The goal is to find k pairwise-disjoint descriptors (one per cluster) such that *all* the objects in S are covered and the total number of tags used in all the descriptors (referred to as cost) is minimized. Refer to example 2.1.3.

Davidson et al. [DGR18] show that even deciding whether there exists a feasible solution is **NP**-hard. They use an Integer Linear Programming (ILP) method to solve the problem (and various relaxed versions which provide useful descriptions even if there is no *exact* feasible solution) on twitter datasets. They point out that this approach gives interesting and representative descriptions for clusters.

1.2 Overview of the Thesis

Davidson et al. [DGR18] proposed an ILP version of the *Cluster Description Problem* [DGR18]. They observed that the size of the ILP is linear with respect to the number of instances, tags, and clusters. However, they found that sometimes the formulation gave inconsistent explanations. Furthermore, they observed that solution often gave highly unbalanced coverage i.e., one cluster gets covered well, but not the others.

Sambaturu et al., [SGD+19] extended the formulation by Davidson et al. [DGR18] to address these issues of unbalanced coverage. They developed a suite of algorithms which give rigorous approximation guarantees for the cluster description problem [DGR18]. They have introduced the *Minimum Constrained Cluster Description Problem* (henceforth, referred to as MinConCD) for cluster description, with *simultaneous* coverage guarantees on all the clusters (defined formally in Section 2.1.1). Informally, given a requirement $M_i \leq |C_i|$ for the number of objects to be covered in each cluster C_i, their goal is to find pairwise-disjoint descriptors of minimum cost such that at least M_ℓ objects are covered in C_ℓ. However, this problem is very difficult. Specifically, if the coverage constraints for each cluster must be met, then unless $\mathbf{P} = \mathbf{NP}$, for any $\rho \geq 1$, there is no polynomial time algorithm that can approximate the cost within a factor of ρ. Therefore, they proposed a notion of (α, δ)–approximate solution which is defined in the following manner: ensuring the coverage of at least αM_ℓ objects in cluster C_ℓ $(1 \leq \ell \leq k)$ using a cost of at most δB^*, where B^* is the minimum cost needed to satisfy the coverage requirements. An algorithm is a factor α approximation for a problem if and only if for every instance of the problem it can find a solution within a factor α of the optimal solution [GGU72, Joh74]. Often, an optimization problem involves several parameters. A bicriteria approximation algorithm achieves a certain approximation ratio while violating some constraint by some bounded amount. For an example of bicriteria approximation algorithm, see [MRS+98].

Sambaturu et al., [SGD+19] have also proposed a randomized algorithm, Round (formally defined in Section 1), for MinConCD, which is based on rounding a linear programming (LP) solution and provides a $(1/8, 2)$–approximation, with high probability. By definition, randomized algorithm is an algorithm which apart from the input takes a source of random number and makes random choices during excecution. For an example, see [MR95].

As part of this thesis, we have explored the following variations of MinConCD:

- A bounded overlap version of the problem for $k = 2$, where cluster descriptors may overlap. In this variation, given input parameters for M_ℓ for $\ell \in [k]$ and overlap limit B_o, the objective is to find a solution of minimum cost such that $|V_\ell(X)| \geq M_\ell$ for each ℓ and $\sum_{\ell \neq \ell'} |X_\ell \cap X_{\ell'}| \leq B_o$. The details are discussed in Section 2.3.1.
- A pair of tags version, where pair of tags are added to the attribute set. The set T of tags is extended to T_{ext} by adding every pair (j, j'), where $j, j' \in T$. The set T_{ext} is then used for finding descriptors. The results are discussed in Section 2.5.

Furthermore, we have empirically evaluated and analyzed MinConCD and Round on various real and synthetic datasets. To achieve this, we designed experiments based on the following research questions:

1. **Performance:** Does Round give solutions with good approximation guarantee in practice? Does it scale to large real world datasets? How do the results of MinConCD compare with that of [DGR18]?
   - **Observations:** Our results indicated the following:
     - The approximation solution of Round was close to the optimum solution and significantly better than the worst-case theoretical guarantees.
     - Round scaled well to instances which were over two orders of magnitude larger than those considered in [DGR18].
   - **Extension:** We explored variations in the suggested rounding scheme. Refer to Sections 2.1.1, 1, 2.5.4 and [SGD$^+$19] for details on formulation and rounding.

2. **Effects of parameters:** What is the effect of cost and coverage parameters on solution quality and feasibility?

   - **Observations:**

     - The spectrum of descriptors obtained by varying cost and coverage parameters allowed a better understanding and exploration of the clusters.
     - Our results confirmed that atleast M_ℓ objects were covered in each cluster. Additionally, our results indicated that solution quality and feasibility also depends on the density of the dataset being considered.

3. **Explanation of clusters:** Do the solutions provide interpretable explanations of clusters in real world datasets?

   - **Observations:**

     - Our results indicated that MinConCD gave insightful results for the Threat dataset [2.4.1]. Refer to Table 2.3 for more details of the dataset. Our results returned a small set of intuitive attributes which separated the harmful sequences from the harmless ones. More details and results of the qualitative analysis can be referred to from Table 2.15.

4. **Pair of tags:** Does pair of tags provide a more precise explanation of the clusters?

   - **Observations:**

     - Pair of tags increased the feasible regime for a few datasets. For a feasible instance it was observed the solutions obtained using T and T_{ext} were quite similar. However, T_{ext} sometimes provided more meaningful descriptions.
     - Pair of tags increased the possibility of an object being covered.
     - Fewer tags provided a reasonable explanation of the clusters.

5. ***k*-cluster:** How do the results depend on k (i.e., when $k = 2$ vs $k = 4$) for Threat dataset [2.4.1]?

    - **Observations:**

        – For Threat dataset [2.4.1], different tags got selected when $k = 2$ and $k = 4$. Details of the Threat datasets can be referred to from Table 2.3. In terms of coverage, $k = 2$ provided better overall coverage.

1.2.1 Why are we doing what we are doing?

There is a considerable amount of work currently being done for cluster summarization using various techniques like Predictive clustering, Constrained clustering, Conceptual clustering, and Conceptual clustering using constrained programming. Conceptual clustering [Fis87] focuses on using a set of features to create the clusters and then uses the same set of features to explain the generated clusters. More work is being done in the field of conceptual clustering with constrained programming [MK10], however, these approaches again focus on explaining the clusters while generating them. Predictive clustering [Lan96] on the other hand uses both predictive modeling and conventional clustering technique to generate clusters and perform predictions. All these techniques focus on explaining the clusters while generating them.

Our objective is very different from all of the above approaches. We aim to explain the clusters after clustering is done, without knowing about the technique that was used to generate the clusters. Moreover, we aim to use the attributes which were not used in the clustering technique to explain the results of the clustering.

Our approach works well for most of the real and synthetic datasets (refer to Section 2.5 for details). However, the approach did not provide expected results for the following scenarios:

- The dataset is extremely sparse. Since our rounding scheme is probabilistic it did not work well for sparse datasets. We observed that cluster coverage dropped significantly in this case.

- The approach did not return expected results when coverage requirement is not constant times the number of objects present in a cluster.

Chapter 2

The Cluster Description Problem

2.1 Preliminaries

2.1.1 Notation and Definition

Let $S = \{s_1, \ldots, s_n\}$ be a set of n objects, and $\pi = \{C_1, \ldots, C_k\}$ be a partition of S into $k \geq 2$ clusters. Let T be the universe of m tags such that each object $s_i \in S$ is associated with a subset $t_i \subseteq T$ of tags. A solution is a subset $X \subseteq T$, and will be represented as a partition $X = (X_1, \ldots, X_k)$, where X_ℓ is the descriptor (i.e., subset of tags) used for cluster C_ℓ. $s_i \in S$ is *covered* by a set $X \subseteq T$ of tags if $X \cap t_i \neq \emptyset$. Let $E(j) = \{s_i : j \in t_i\}$ be the set of all objects that can be covered by the tag $j \in T$. Let $\eta = \max_j |E(j)|$ denote the maximum number of objects covered by any tag in T. Let $\gamma = \max_i |t_i|$ denote the maximum number of tags associated with any object in S. Objects $s_i, s_{i'} \in S$ are said to be *dependent* if $t_i \cap t_{i'} \neq \emptyset$, i.e., if their tag sets overlap. Let $\Delta(i) = |\{i^{'} : t_i \cap t_{i'} \neq \emptyset\}|$ denote the degree of dependence of s_i, and let $\Delta = \max_i \Delta(i)$ be the maximum dependence. Finally, for a solution $X = (X_1, \ldots, X_k)$, let $V_\ell(X) = \{s_i \in C_\ell : t_i \cap X_\ell \neq \phi\}$ be the subset of objects in C_ℓ covered by X, $1 \leq \ell \leq k$. For an integer k, we use $[k]$ to denote the set $\{1, \ldots, k\}$. The Table 2.1 summarizes the above stated notations.

Notation	Definition
S	A set of n objects
n	Number of objects
π	A set of k clusters
k	Number of clusters
C_ℓ	Cluster (ℓ denotes cluster number)
T	Universal set of tags
s_i	An object (or, a data point)
X	Solution set $(X_1, \dots, X_\ell)$
$E(j)$	A set of all objects covered by any tag j
η	Maximum number of tags associated with any object
Δ	Degree of dependence of an object.

Table 2.1: This table summarizes the notations and their definitions, as discussed in the previous section.

2.1.2 Problem Statement

The objective is to find a solution X that *simultaneously* ensures high coverage $V_\ell(X)$ in each cluster C_ℓ. An obvious choice would be to consider a max-min type of objective $X = \operatorname{argmax} \min_\ell |V_\ell(X)|$ (see, e.g., [Udw18]). However, this doesn't allow handling domain specific requirements (e.g., higher coverage for the cluster of threat sequences in genomic data). Therefore, a more general formulation is considered by Sambaturu et al. [SGD+19]; wherein they specify a coverage requirement for each cluster.

The Minimum Constrained Cluster Description (MinConCD) problem

Instance: A set $S = \{s_1, \dots, s_n\}$ of objects, a partition $\pi = \{C_1, \dots, C_k\}$ into $k \geq 2$ clusters, a universe T of m tags, set t_i for each object s_i, M_ℓ for each cluster C_ℓ.

Objective: To find a solution $X = (X_1, \dots, X_k)$ that minimizes the cost $\sum_{\ell=1}^{k} |X_\ell|$ satisfying the following conditions:

1. The subsets in X are pairwise-disjoint,

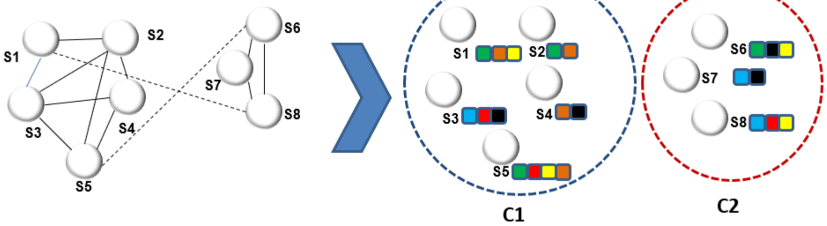

Figure 2.1: Pictorial representation of the toy dataset. There are 5 data-points in Cluster 1 & 3 data-points in Cluster 2. The tags are denoted by colors.

2. For each cluster C_ℓ, $|V_\ell(X)| \geq M_\ell$

2.1.3 Example: Toy Dataset

Table 2.2 describes the toy dataset and Figure 2.1 shows the pictorial representation of the 2 clusters. Re-iterating what's been defined above:

Dataset	$\|S\|$	$\|T\|$	$\|C_1\|$	$\|C_2\|$
Toy	8	6	5	3

Table 2.2: Description of toy dataset

Let, $S = \{s_1, \ldots, s_n\}$, $n = 8$ be the set of objects.

Let, $\pi = \{C_1, \ldots, C_k\}$ be a partition of S into $k = 2$ clusters.

Let, $T = \{green, brown, yellow, blue, red, black\}$ be the universe of m tags, where $m = 6$.

Each object $s_i \in S$ is associated with a subset $t_i \subseteq T$ of tags and $\gamma = max_i|t_i|$.

Hence, $t_1 = \{green, brown, yellow\}$, $t_2 = \{green, brown\}$, $t_3 = \{blue, red, black\}$, $t_4 = \{brown, black\}$, $t_5 = \{green, red, yellow, brown\}$, $t_6 = \{green, black, yellow\}$, $t_7 = \{blue, black\}$, $t_8 = \{blue, red, yellow\}$, $\gamma = 4$.

Let, $E(j)$ be the set of all objects that can be covered by tag $j \in T$ and $\eta = max_j|E(j)|$.

Hence, $E(red) = \{s_3, s_5, s_8\}$, $E(green) = \{s_1, s_2, s_5, s_6\}$, $E(yellow) = \{s_1, s_5, s_6, s_8\}$, $E(brown) =$

$\{s_1, s_2, s_4, s_5\}$, $E(blue) = \{s_3, s_7, s_8\}$, $E(black) = \{s_3, s_4, s_6, s_7\}$, $\eta = 4$.

Let, $\Delta(i) = |\{i' : t_i \cap t_{i'} \neq \emptyset\}|$ denote the degree of dependence of s_i. Hence, $\Delta(1) = 5$, $\Delta(2) = 3$, $\Delta(3) = 5$, $\Delta(4) = 6$, $\Delta(5) = 6$, $\Delta(6) = 6$, $\Delta(7) = 5$, $\Delta(8) = 4$. Let, $\Delta = \max_i \Delta(i)$ be the maximum dependence. Hence, $\Delta = 6$.

A *solution* is a subset $X = (X_1, X_2)$ where $X \subseteq T$. MinConCD gives: $X_1 = \{brown, blue\}$, and $X_2 = \{yellow, black\}$. Therefore, $V_1(X) = \{s_1, s_2, s_3, s_4, s_5\}$.

It can be observed that: $t_1 \cap X_1 \neq \emptyset$ and likewise for all $s_i \in V_1(X)$. Similarly, $V_2(X) = \{s_6, s_7, s_8\}$ and, for all $s_i \in V_2(X)$, $t_i \cap X_2 \neq \emptyset$

2.2 Algorithm Round: Approximation using Linear Programming and Rounding

Since MinConCD is **NP**-hard [DGR18], Sambaturu et al., explored approximation algorithms [SGD+19]. They proposed that a solution X is an (α, δ)-approximation if: (1) for each cluster C_ℓ, $|V_\ell(X)| \geq \alpha M_\ell$, and (2) $\sum_\ell |X_\ell| \leq \delta B$. Their approach for approximating MinConCD is based on Linear Programming (LP) relaxation and rounding the fractional solution. This is a common approach for many combinatorial optimization problems, including the problems that have covering constraints (see, e.g., [WS11]). However, the disjointness requirement for descriptors poses a challenge in terms of dependencies and requires a new approach. They have described two cases, namely (1) when $M_\ell = \Theta(|C_\ell|)$ and (2) when M_ℓ is arbitrary. The rounding methods and analysis are different in these cases. Our experiments for this thesis consider case (1).

2.2.1 LP Relaxation.

For this problem if there exists an LP which can solve the problem then by definition **P=NP**. Hence, we have formulated an ILP and then relaxed it to an LP. For each $j \in T$, $\ell \in [k]$, $x_\ell(j)$ is an indicator variable, which is 1 if tag $j \in T_\ell$. There is an indicator variable $z(i)$ for $s_i \in S$, which is 1 if object s_i is covered. The following LP ($\mathcal{P}$) is considered, in which all variables are relaxed (from being binary) to be in $[0,1]$.

$$\begin{aligned}
\min \sum_{\ell=1}^{k} \sum_{j} x_\ell(j) & & \text{s.t.} \\
\forall \ell,\ \forall s_i \in C_\ell : \sum_{j \in t_i} x_\ell(j) & \geq & z(i) \\
\forall \ell : \sum_{s_i \in C_\ell} z(i) & \geq & M_\ell \\
\forall j : \sum_{\ell} x_\ell(j) & \leq & 1 \\
\text{All variables} & \in & [0,1]
\end{aligned}$$

2.2.2 Rounding Algorithm

Algorithm 1 describes the steps of Round. The linear program $\mathcal{P}$ can be solved (Step 1) using standard techniques in polynomial time, e.g., [KT06], a fractional solution to the variables of $\mathcal{P}$ can be obtained efficiently whenever there is a feasible solution. We have analysed the performance of algorithm Round in 2.5 by Theorem 2.2. The details of the theorem can be referred to from [SGD+19]. The algorithm is defined to run for any number of clusters where k is the total number of clusters. Notation ℓ represents cluster number. For instance, if $k = 2$ (which means there are 2 clusters under consideration); for cluster 1, $\ell = 1$ and cluster 1 is denoted as C_1. Similarly for cluster 2, $\ell = 2$ and cluster 2 is denoted as C_2.

Algorithm 1: Algorithm Round

```
Input  : $S$, $n$, $\pi = \{C_1, \ldots, C_k\}$, $T$, $M_\ell$ for each $\ell = 1, \ldots, k$, $n_{iterations}$
Output: $X = (X_1, \ldots, X_k)$
Solve $\mathcal{P}$. If it is not feasible, return "no feasible solution". Else, let $x^*, z^*$ denote the
  optimal fractional solution and $B$ denote the associated cost.
For all $j$, and for all $\ell$, set $x_\ell(j) = x^*_\ell(j)/2$, and for all $s_i$ set $z(i) = z^*(i)/2$.
for $n_{iterations}$ times do
    for $j \in T$ and $\ell = 1, \ldots, k$ do
        With probability $x_\ell(j)$, round $X_\ell(j) = 1$ and $X_{\ell'}(j) = 0$ for all $\ell' \neq \ell$.
        With probability $1 - \sum_\ell x_\ell(j)$, set $X_{\ell'}(j) = 0$ for all $\ell'$.
    end
    for $s_i \in S$ do
        If $X_\ell(j) = 1$ for some $j \in t_i$, define $Z(i) = 1$.
    end
    For each $\ell$, define $Z_\ell = \sum_{s_i \in C_\ell} Z(i)$.
    If $Z_\ell \geq M_\ell/8$ for each $\ell$, and $\sum_\ell \sum_j X_\ell(j) \leq 2B$, return $X$ as the solution and stop.
end
```

2.3 Other Variations

The problem considered in [DGR18] was to maximize $\sum_\ell |V_\ell(X)|$, i.e., the total number of objects covered—referred to as the MCBC problem. Another variation referred to as MinConCDO allows limited overlap between descriptors for different clusters. Given input parameters M_ℓ for $\ell \in [k]$, budget B and overlap limit B_o, the objective is to find a solution $X = (X_1, \ldots, X_k)$ such that for each ℓ, $|V_\ell(X)| \geq M_\ell$, $\sum_\ell |X_\ell| \leq B$, and $\sum_{\ell \neq \ell'} |X_\ell \cap X_{\ell'}| \leq B_o$.

2.3.1 Bounded Overlap Version

For this variant case where $k = 2$ is considered. The following changes are done to the LP $(\mathcal{P})$ as discussed in Section 2.2:

- For each $j \in T$, there is an indicator variable $y(j)$, which is 1 if j is common to the descriptors X_1 and X_2.

- For each j, constraint $x_1(j) + x_2(j) \leq 1$ is replaced by $x_1(j) + x_2(j) \leq 1 + y(j)$.
- An additional constraint $\sum_j y(j) \leq B_o$ has been added.

This rounding is a modification of Step 3 of algorithm Round.

- Let x, y, z be the fractional solution obtained by scaling down the optimal solution by a factor of 2 (as in Step 2).
- Replace Step 3(a) with the following steps:
  - With probability $y(j)$: $X_1(j) = 1, X_2(j) = 1$
  - With probability $x_1(j) - y(j)$: $X_1(j) = 1, X_2(j) = 0$
  - With probability $x_2(j) - y(j)$: $X_1(j) = 0, X_2(j) = 1$
  - With probability $1 - x_1(j) - x_2(j) + y(j)$: $X_1(j) = X_2(j) = 0$
- Let $Z(i)$ and Z_ℓ be as defined in Step 3(b). The algorithm ends if the following conditions hold: $Z_\ell \geq M_\ell/8$ for $\ell = 1, 2$, $|X_1| + |X_2| \leq 3B$, and $|X_1 \cap X_2| \leq 3B_o$.

Explanation: Round

In this section, Round which is a dependent rounding scheme is explained using an example: Let $k = 2$

- For each tag j:
  - With probability $x_1(j)$, round $X_1 = 1$ and $X_2 = 0$.
  - With probability $x_2(j)$, round $X_1 = 0$ and $X_2 = 1$.

  - With probability $1 - x_1(j) - x_2(j)$, round $X_1 = 0$ and $X_2 = 0$.

- The object s_i is covered (i.e., $Z(i) = 1$) if atleast one tag from t_i is selected.
- If atleast $M_\ell/8$ objects are covered and not more than $2B$ tags are selected, the algorithm stops.

Limitation: Algorithm Round might not return a feasible solution in 1 iteration.

Rounding Detailed Steps: Let $k = 2$

- For each tag j:
  - Calculate cumulative probabilities for tag j.
  - Generate random number $r \in [0, 1)$.
  - Find an interval for r from cumulative probability and assign cluster.

Example:

- Let the fractional solutions be [0.2 0.3].
- Then, the cumulative probabilities will be [0.2 0.5 1.0].
- Generate random number r.
- Find interval for r from the cumulative probability.
  - If $0 \leq r < 0.2$; then tag will get assigned to cluster C_1.
  - If $0.2 \leq r < 0.5$; then tag will get assigned to cluster C_2.
  - If $0.5 \leq r < 1.0$, then tag will not be assigned to any cluster.
- The algorithm may not converge in 1 iteration hence, execute all above steps many times.

- The object s_i is covered (i.e., $Z(i) = 1$) if atleast one tag from t_i is selected.
- If atleast $M_\ell/8$ objects are covered and not more than $2B$ tags are selected, the algorithm returns the solution.

Lemma 2.1. *For each $\ell \in [k]$, the expected number of objects covered in cluster C_ℓ by a solution X in any round of Step 4 of algorithm* Round *is at least $M_\ell/4$ [SGD+19].*

Theorem 2.2. *Suppose an instance of MinConCD satisfies the following conditions: (1) $M_\ell \geq a|C_\ell|$ for all $\ell \in [k]$, and for some constant $a \in (0, 1]$, and (2) $(\Delta + 1) \leq \min_\ell \frac{d|C_\ell|}{\log n}$ and $d \leq \frac{a^2}{576}$, and (3) $k \leq n/4$. If the LP relaxation $(\mathcal{P})$ is feasible, then with probability at least $1 - \frac{1}{n}$, algorithm* Round *successfully returns a solution X, which is a $(1/8, 2)$–approximation [SGD+19].*

Since we have used theorem 2.2 for empirical analysis of Round, I am re-iterating it intuitively. Assume,

1. Coverage requirement is atleast constant times the number of objects in the cluster.
2. $(\Delta + 1) \leq \min_\ell \frac{d|C_\ell|}{\log n}$ and $d \leq \frac{a^2}{576}$, where Δ is the maximum degree of dependence.
3. the number of clusters is at most $n/4$ where, n is total numbers of objects.

Suppose all of the above assumptions are satisfied and if LP relaxation is feasible the algorithm Round returns a solution with high probability, which is a $(1/8, 2)$-approximation.

Proof of correctness: Round

The formal proof of the algorithm Round can be referred to from [SGD+19]. This section re-iterates the proof:

The objective of Minimum Constrained Cluster Description (MinConCD problem):

(1) The subsets in X are pairwise-disjoint, and

(2) for each cluster C_ℓ, $|V_\ell(X)| \geq M_\ell$

To assert objective (1), algorithm Round uses probabilistic rounding. For example, when $k = 2$, X_1 and X_2 are the two solutions and the objective is to ensure X_1 and X_2 are pairwise-disjoint.

In each iteration of Round, for each tag and for each cluster, with probability $x_l(j)$ we set $X_1 = 1$ and $X_2 = 0$, and with probability $x_2(j)$ we set $X_2 = 1$ and $X_1 = 0$.

With probability $1 - x_1(j) - x_2(j)$ set $X_1 = 0$ and $X_2 = 0$.

This approach ensures that the solution obtained after rounding scheme (X_1 and X_2) are pairwise-disjoint.

The second objective of MinConCD is to ensure for each cluster $|V_\ell(X)| \geq M_\ell$. The correctness of this objective follows from Theorem 2.2 which states that Round gives $(1/8, 2)$-approximation. To ensure this, Round satisfies that $|V_\ell(X)|$ is atleast $\geq M_\ell/8$ and violates the budget constraint by 2 by allowing B to be at most $2B$.

Implementation Details

We have used Gurobi solver in Python to implement "MinConCD" and "Round". Gurobi solver is a commercial Mathematical Programming solver which is used to run LP and ILP models. For complex models when constrains and decision variables grow in size, Gurobi solver facilitates parallel computation and output generation within the available time. The experiments were run in parallel on high performance computing clusters with maximum cores per job: 16, maximum memory per core: 62 Gb and maximum memory per node per Job: 975 Gb. The gurobi code can be referred to from "https://github.com/Aparna-Gupta/thesis_code_AparnaGupta".

2.4 Experiments

2.4.1 Datasets

Table 2.3 provides a summary of all the datasets being used for experiments and analysis.

Dataset	$\|S\|$	$\|T\|$	$\|C_1\|$	$\|C_2\|$
Genome (Threat)	248	4632	73	175
Uniref90	21537	2193	13406	8131
Flickr	2454	175	1052	1402
Philosophers	240	14000	102	138
Synthetic 1 ($p = 0.05$)	100	100	48	52
Synthetic 2 ($p = 0.2$)	100	100	58	42
Synthetic 3 ($p = 0.05$)	1000	1000	502	498
Synthetic 4 ($p = 0.1$)	1000	1000	478	522
Synthetic 5 ($p = 0.15$)	1000	1000	479	521
Synthetic 6 ($p = 0.2$)	1000	1000	497	503

Table 2.3: Description of real-world and synthetic datasets used for experiments. $|S|$ denotes the total number of objects. $|T|$ is the tagset. $|C_1|$ is the total number of objects in cluster 1 and $|C_2|$ is the total number of objects in cluster 2.

FunGCAT Dataset

An increasing number of nucleotide and amino acid sequences and their associated attributes that specify their interaction with other biological entities are now available in modern biological databases. The notion of what classifies a "threat" is context-dependent. A gene may be a toxin, a target for antibiotic resistance, an effector, or simply come from a pathogen of concern. As part of IARPA's FunGCAT initiative, threat bins were established using expert manual curation of selected genes. Sequences were selected and annotated to represent a spectrum of perceived threat on an increasing scale of 1-4. The Threat and Uniref90 datasets [JK18, RW18] contain genome sequences and information that may indicate a given gene's threat potential. Uniref90 is the complete version of the Threat dataset. The clusters for these datasets correspond to gene sequences that are *harmful* or *harmless*.

Other Datasets

Flickr Dataset

The Flickr dataset [YML13] consists of images as nodes and relationships between images as edges. A relationship could correspond to images being submitted from the same location, belonging to the same group, or sharing common tags, etc. We use the Louvain algorithm in Networkx [HSSC08] to generate communities of images, and pick the two largest communities as clusters. User defined tags, such as "dog," "person," "car," etc., are provided for each image.

Philosophers Dataset

The Philosophers dataset [YML13] consists of Wikipedia articles on various philosophers. The tags corresponding to each object are the non-philosopher Wikipedia articles to which there is an outlink from the philosopher article. The clusters in the Philosopher dataset are generated by grouping communities that share a common keyword as a single cluster.

Synthetic Dataset

In the synthetic datasets, an object is associated with a descriptor with probability p. A high probability indicates dense matrix.

2.4.2 Experimental Setup

Coverage	**Range of B values**
90%	5, 10, 15, 20, 25, 30
80%	5, 10, 15, 20, 25, 30
70%	5, 10, 15, 20, 25, 30
60%	5, 10, 15, 20, 25, 30
50%	5, 10, 15, 20, 25, 30

Table 2.4: Basic experimental setup used in all research questions. M_ℓ/C_ℓ is the coverage percentage per cluster and B is the cost. Every experiment uses M_ℓ and each B as a parameter.

Table 2.4 explains the experimental setup used for most of the experiments. Column coverage explains the various coverage requirement values that are used throughout the experiments. In our experiments, coverage requirement and cost are the two independent variables and LP/ILP solution is the dependent variable.

Every research question uses this setup as a base along with modifications (either in datasets or in parameter space).

2.4.3 Research Questions

Dataset	$\|S\|$	Coverage
Genome (Threat)	248	216
Uniref90	21537	15845
Flickr	2454	NA
Philosophers	249	NA

Table 2.5: Coverage for a fixed cost ($B = 5$) and fixed coverage percentage (90%). The algorithm did not return a feasible solution for a few datasets.

As part of this thesis we have studied the following research questions:

1. **Effects of parameters:**

- Table 2.5 shows coverage across various real-world datasets for a fixed cost ($B = 5$) and fixed coverage requirement ($M_\ell = 0.9|C_1|$ and $M_\ell = 0.9|C_2|$).
- For a few datasets, lowering the cost did not return a feasible solution. This lead to the following questions:
  - What are the effects of cost and coverage requirements on solution quality and feasibility?
- How does MinConCD compare with the algorithm given in [DGR18].
- What happens if a minimum tag overlap is allowed (Bounded Overlap version)?

2. **Pair of tags:**

   Research Question 1 focuses on the effect of parameters (coverage requirement and cost) on the solution feasibility. A natural question which arises here is, if these are the only factors which affect the solution feasibility? **Can an increase in the number of tags increases the likelihood of getting a feasible solution?**

   To answer the above question we extended the set of tags T to add pair of tags which increased the tag set for each cluster and thereby increased the possibility of an element s_i getting hit. Refer Table 2.6 for new tagset details. The objective here is to understand whether a pair of tags can provide a cleaner and more precise explanation of real-world datasets.

Dataset	$\lvert S\rvert$	$\lvert T\rvert$	$\lvert T_{ext}\rvert$
Threat (Genome)	248	4632	1579754
Flickr	2452	175	10010
Philosophers	240	14000	193483

Table 2.6: $|T_{ext}|$ is the number of tags in the new tagset for real-world datasets. We extend the set of tags T to T_{ext} by adding every pair (j, j'), where $j, j' \in T$, and use T_{ext} for finding descriptions.

Hence, the question that we are looking to answer here is: **Does a pair of tags**

provide a more precise explanation of the clusters?

3. ***k*-cluster Analysis:**

   The two research questions discussed above focus on finding answers with $k = 2$. In this section we aim to understand and compare results when $k \geq 2$. However, due to lack of datasets with objects evenly distributed across multiple clusters, the experiments in this section are focused on $k = 4$ for Threat dataset. The question that we seek to answer here is: **Are results comparable when $k = 2$ vs $k = 4$?**

4. **Performance:**

   So far we ran experiments to analyze the output of algorithm Round. This research question will focus on understanding the performance of algorithm, Round, on various real-world & synthetic datasets. We will analyze the performance comparison between ILP & LP in terms of runtime. Hence, the research questions that we will focus on are:

   - **Does Round give solutions with good approximation guarantees in practice?**
   - **Scalability: does it scale to large real world datasets?**

5. **Explanation of clusters:** This research question presents a qualitative analysis of the results. Once the results are obtained, a detailed analysis is done after consulting the experts. Hence, the research question that we will focus on is:

   - **Do the solutions provide interpretable explanations of clusters in real world datasets?**

2.5 Results

2.5.1 Effects of parameters

Experimental setup: Since the objective of the formulation by Sambaturu et al., [SGD+19] is to minimize the tagset selected and simultaneously maximize the coverage in each cluster, we have used the experimental setup from Table 2.4 to study the effect of various parameters on algorithm Round:

Initial Analysis: To analyze the effect of parameters on algorithm Round, we initially ran experiments on full datasets. Full datasets correspond to all objects and descriptors in the dataset. It was observed that there were a few tags which covered 80-90% of all the objects and hence for any selected parameter space, same tags were being returned. This made it harder to understand which tags were significant to explain each of the clusters. To overcome this issue, the tags which covered more than 2 objects were kept and the rest were removed from the tagset. This approach resulted in a reduced tagset for large datasets.

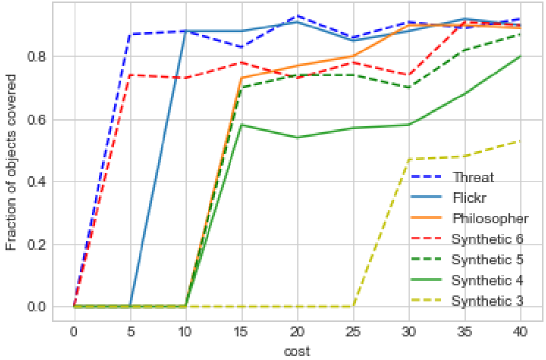

Figure 2.2: Fraction of objects covered (y-axis) vs cost (x-axis) for different datasets (90% fixed coverage). Higher is better. The synthetic datasets increase in density as their number increases. The curves plateau out (indicating the maximum objects that can be covered), with the curves for real datasets growing much faster.

Observations: Figure 2.2 shows that as cost is increased, the chances of finding a feasi-

Coverage	5	10	15	20	25	30	35	40
80%	NA	NA	NA	NA	527	529	627	612
70%	NA	NA	NA	NA	496	549	554	568
60%	NA	NA	NA	492	487	569	572	573
50%	NA	NA	337	454	516	560	666	635

Table 2.7: Table showing the effect of varying coverage requirement over various cost. It is observed that as coverage requirement is relaxed, the dataset becomes easier to cover. This Table is generated for Synthetic3 (extremely sparse) dataset.

ble solution also increased. It shows the fraction of objects covered as a function of the cost. **An increase in cost did increase solution feasibility.** As more tags got picked, the probability of more objects getting hit also increased, thereby increasing the solution feasibility.

To further investigate the effects of parameters on solution feasibility we ran multiple experiments on datasets with sparse and dense data matrices. The dataset description for such datasets can be referred to from Table 2.3 and 2.4.1. **For synthetic datasets, as density increases, they become easier to cover**. As density increases, the number of tags associated with an object also increase, thereby increasing the chance of an object getting hit.

As density increases, number of tags associated with an object also increases. Furthermore, increasing the chances of an object getting hit. Hence, we can say that for datasets with dense data matrices LP returned a feasible solution even at a lower cost.

Table 2.17 supports our claim that as coverage requirement is relaxed, an extremely sparse dataset becomes easy to cover. The results are not specific to a particular parameter space instead depend on the dataset (how dense or sparse it is) under consideration. For example, when $B = 10$ and 90% fixed coverage is required, synthetic dataset 3 (Synthetic 3) does not return a feasible solution whereas synthetic dataset 6 (Synthetic 6) return a feasible solution. If the coverage requirement is relaxed from 90% to 60%, at $B = 10$, Synthetic 3 returns a feasible solution.

Comparison of algorithm Round **with [DGR18]:** The exact coverage formulation of [DGR18] (which corresponds to $M_\ell = |C_\ell|$ for all ℓ) is infeasible for some of the datasets we consider. Instead, we examine the cluster descriptions computed using the cover-or-forget formulation of [DGR18], which maximizes the total number of objects covered. Figure 2.3 show the coverage percent for each cluster, i.e., $(|V_\ell(X)|/M_\ell) \times 100\%$, ($y$-axis) versus the cost of the solution (x-axis), for the threat dataset. It shows that the coverage is highly imbalanced. For instance, with 4 tags, almost 75% of elements in cluster C2 are covered, whereas only 10% of elements in cluster C1 are covered. This is a limitation of the cover-or-forget approach. The cluster specific coverage requirements in MinConCD can help alleviate this problem. Figure 2.4 shows the result of MinConCD. We can observe that both clusters have balanced coverage.

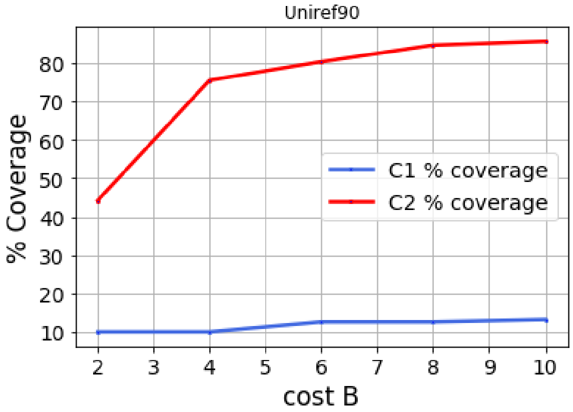

Figure 2.3: Coverage % in each cluster (y-axis) vs cost (x-axis) [Threat dataset] for the formulation by [DGR18]. The plot shows that coverage in 2 clusters (C_1 & C_2) is highly imbalanced. This plot was generated for Uniref90 threat dataset.

The above experiments strictly follow the disjointness condition. We conducted a similar set of experiments for bounded overlap variant of the problem by varying B_o from 1 to 3 for lower B and 5 for higher B. It was observed that the number of tags being picked dropped

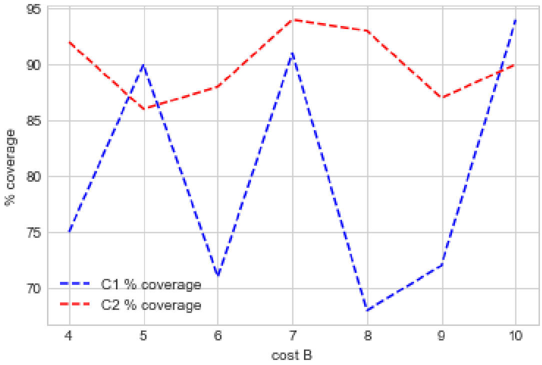

Figure 2.4: Coverage % in each cluster (y-axis) vs cost (x-axis) [Threat dataset] for the formulation by Sambaturu et al. The plot shows that the approach (MinConCD) suggested by Sambaturu et al., [SGD+19] overcomes the cluster imbalance issue.

significantly and very few tags could explain both clusters reasonably well. The variant gave a feasible solution for lower B.

2.5.2 Pair of Tags

Experimental Setup: For pair of tags we have used the experimental setup defined in Table 2.4. The datasets used for these experiments are specified in Table 2.6. Also, for these experiments the extended tagset (T_{ext}) has been used instead of the base tagset (T). Here, we discuss the impact of adding a pair of tags to the set of attributes. A pair of tags (j, j') is considered as an attribute that covers only those objects covered by both the tag j and j'.

Initial Analysis: In this section we will talk about initial results for the flickr dataset. The tagset size increased from 175 to 10100. The pairs which covered atleast 2 objects were kept and the remaining were dropped. This reduced the extended tagset size considerably. Next section gives an overview of our findings and presents a qualitative analysis of the results obtained from flickr dataset.

Observations and Discussions: We ran rounding algorithm 1 on extended tagset and no-

Single attributes	Pairs of tags
“tree”	“tree, night”
“night”	“clouds, night”
“river”	“Neutral Illumination”
“clouds”	“Motion Blur”
“male”	
“Outdoor”	
“Neutral Illumination”	

Table 2.8: Flickr dataset: Tags selected for cluster one (C_1) when single tags and extended tagset was used. The column on the left gives a brief overview of the commonly selected single tags for cluster 1 and, the column on the right gives a brief overview of the commonly selected tag for cluster 1 from extended tagset.

Single attributes	Pairs of tags
“people”	“female, people”
“small group”	“river, people”
“No blur”	“road, people”
“road”	“nighttime, bridge”
“female”	
“animal”	

Table 2.9: Flickr dataset: Tags selected for cluster two (C_2) when single tags and extended tagset was used. The column on the left gives a brief overview of the commonly selected single tags for cluster 2 and, the column on the right gives a brief overview of the commonly selected tag for cluster 2 from extended tagset.

ticed that **in some cases, for a smaller budget, extended tagset returned a feasible solution while it was infeasible with just the base set of attributes. This is due to an increase in the number of tags for the algorithm to pick from for explaining the clusters.** In most other cases, the number of tags picked when extended tagset was used as input is close to that used with the base set of attributes. **The pair of tags in spite of not reducing the description cost seems to provide more meaningful descriptions.** The reason why description cost did not increase is that every pair of the tag selected is charged a cost of 1 instead of 2.

For instance, experiments on Flickr dataset with a similar setting (like that of single tagset) produced similar results. The pair of tags which got picked were mostly the combination of single tags. For example, (“tree” , “night” ,“river”, “clouds”, “male”, “Outdoor”, “Neutral Illumination”) are some of the tags which explain the images in cluster 1 and (“tree, night” , “clouds, night”, “Neutral Illumination”, “Motion Blur”) are the corresponding pair of tags picked. Therefore, we can say that **with the pair of tags a lesser number of tags could provide a reasonable explanation of the cluster.**

This is a stricter setting since only the presence of a combination of tags determine a tag’s presence. The advantage of using this approach is that we did not have to make changes in our LP formulation and the code was run as it is. **Future work** for this formulation can be to charge every pair of tags selected as two instead of one (as stated above) and either single tag or a pair of tags (containing that single tag) gets selected.

2.5.3 k-cluster Analysis

Experimental Setup: For k-cluster analysis we have used Genome (Threat) and Uniref90 datasets. Table 2.10 shows the number of objects per cluster (when $k = 4$). The type of experiments run can be referred to from Section 2.4. Threat level (referred to as threat bin $1 - 4$) were used to partition the dataset into clusters. For $k = 4$, threat bins $[1 - 4]$ are used as individual clusters.

Dataset	$\|S\|$	$\|C_1\|$	$\|C_2\|$	$\|C_3\|$	$\|C_4\|$
Genome (Threat)	248	35	38	65	110
Uniref90	21537	494	12912	7776	355

Table 2.10: The Table shows the number of objects per cluster for Genome (Threat) & Uniref90 dataset. This Table was used to analyze the number of objects covered per cluster when experiments were run for $k = 4$. The qualitative analysis is done for Genome dataset, the results of which are discussed in the Qualitative Analysis section.

Initial Analysis: Due to lack of datasets with high density per cluster, we analysed this approach on Threat dataset. As described above, Threat dataset contains four threat levels where 1 is no threat to 4 being the highest threat. For this experiment we considered each threat label as a cluster. Thus resulting in 4 clusters. Now, we discuss the comparison of results when $k = 2$ vs $k = 4$ for Threat dataset. **It is observed that with $k = 4$ most of the tags picked were different from tags when $k = 2$.** Figure 2.5 below shows the comparison of total coverage when $k = 2$ vs $k = 4$ for Threat dataset.

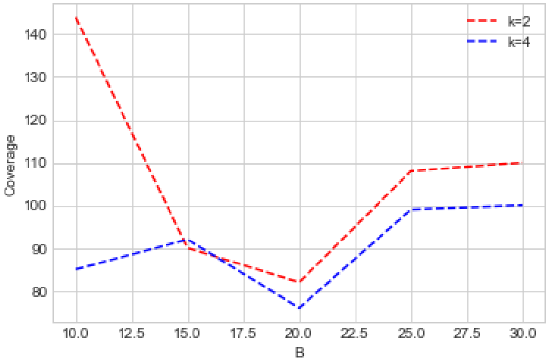

Figure 2.5: Coverage (y-axis) vs budget (x-axis) for $k = 2$ & $k = 4$, where k denotes the number of clusters. This plot was generated using Threat (Genome) dataset. Coverage is higher when the number of clusters selected is 2.

Qualitative Analysis:

This section presents a qualitative analysis of the results obtained when experiments were run for $k = 2$ and $k = 4$ clusters.

Cluster Explanation ($k = 2$):

- Results for Cluster C_1:
  - GO:0003824 (“catalytic activity”), GO:0065007 (“biological regulation”), GO:0005623 (“cell”), GO:0044237 (“cellular metabolic process”).
- Results for Cluster C_2:

  - KW-0800 (toxin), 155864.Z3344 ("Shiga toxin 1"), IPR011050 ("Pectin lyase fold/virulence"), and IPR015217 ("invasin domain"), KW-0732 ("signal peptide"), KW-0614 ("plasmid"), KW-0964 ("secreted"), and GO:0050896 ("response to stimulus").

Cluster Explanation ($k = 4$):

- Results for Cluster C_1:
  - GO:0009058 ("chemical reactions and pathways resulting in formation of substances"), GO:0042895 ("antibiotic transmembrane transporter activity"), IPR016129.
- Results for Cluster C_2:
  - IPR000734 ("Triacylglycerol lipase family"), GO:0000041 ("The directed movement of transition metal ions into, out of or within a cell, or between cells")
- Results for Cluster C_3:
  - IPR019553 ("toxin"), UPI0000136BBC ("Sea anenome toxin"), IPR015917 ("caspase. Involved in apoptosis. Possibly a minor threat"), KW-1222 ("toxin").
- Results for Cluster C_4:
  - GO:0090729 ("good indicator of toxin"), ENOG410XQE6 ("Intimin/invasin"), UPI00001700A1 ("Shiga-like toxin"), GO:0046931 ("partial indicator of threat"), UPI0000520D62 ("toxin"), COG3210 ("A virulence factor").

2.5.4 Performance

Experimental Setup:

To measure the performance of algorithm Round we ran experiments based on the experimental setup explained in Section 2.4. The optimal solution is obtained from ILP and the approximate solution is obtained by dividing the solution obtained from Round by the Optimal solution.

Observations: First, we consider the approximation guarantee of Round in practice.

Optimal Solution	Cost	Round	Approximation ratio
232	5	216	0.93
247	10	217	0.87
248	15	207	0.83
248	20	232	0.93
248	25	214	0.86
248	30	227	0.91
248	35	221	0.89
248	40	229	0.92

Table 2.11: Optimal Solution (ILP) and approximate Solution for various costs. Genome (Threat) dataset. 90% fixed coverage. The approximation ratio is calculated using the formula Round/Optimal Solution.

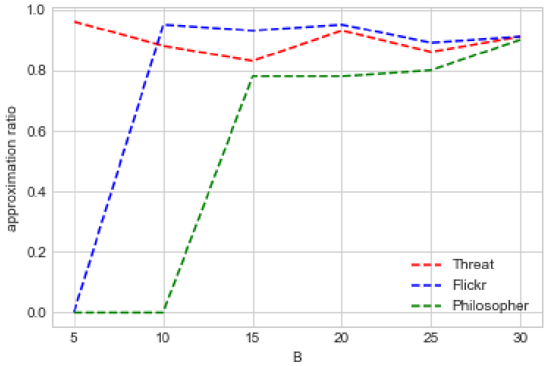

Figure 2.6: Approximation ratio of Round (y-axis) vs cost (B) (x-axis) for different real-world datasets (higher is better). As the cost is increased, datasets become easier to cover.

Figure 2.6 shows the approximation ratios (i.e., the ratio of the number of objects covered by the solution computed using Round, to that of an optimum solution) for different datasets. The analysis in Theorem 2.2 only guarantees a small constant factor, but plot shows that the approximation factors is always more than 0.8, and more than 0.9 in most cases. This suggests that Round gave solutions which were very close to the optimal. Note that the curves are non-monotone—this is due to the stochastic nature of Round.

Experimental Setup and Observations for Variations in Rounding:

- We did not scale down $x_1(j)$, $x_2(j)$ and z_i by a factor of 2 as mentioned in Round. It was observed that not scaling down the fractional values by 2 provided good coverage for each cluster.

- In this variation we did not scale down $x_1(j)$, $x_2(j)$ and z_i by a factor of 2 and deterministically rounded fractional values which were large i.e., tags with fractional values ≥ 0.5 were rounded to 1.

For empirical analysis we ran experiments on variations of rounding where we tried following variations. We observed that for higher budget Round gave higher overall as well as clusterwise coverage. Refer to figure 2.7.

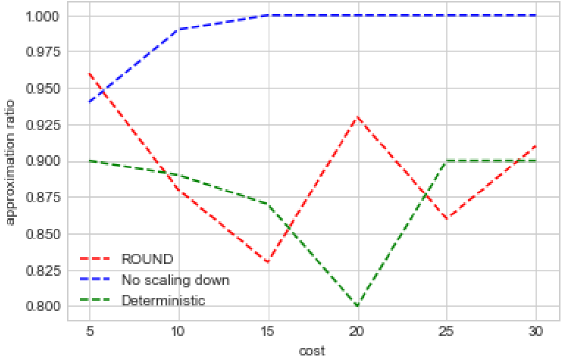

Figure 2.7: Coverage (x-axis) vs B (y-axis) with variations in Rounding for Threat dataset. 90% fixed coverage. Round is the proposed algorithm. In no scaling down, LP fractional solution is not scaled down by 2. In deterministic rounding scheme, the values greater than 0.5 are rounded to 1 and less than 0.5 are rounded to 0.

Experimental Setup: Tradeoffs between α and δ:

We varied the values of α and δ to empirically analyze the tradeoffs between them. Following is the experimental setup for Philosophers dataset:

1. For 90% fixed coverage and $\alpha = 2$: set δ values to $\frac{1}{4}$, $\frac{1}{2}$, $\frac{1}{3}$.
2. For 90% fixed coverage and $\delta = \frac{1}{8}$: set α values to $3, 4, 5$.
3. For 90% fixed coverage: set $\alpha = 3$ and $\delta = \frac{1}{4}$.

Observations: For point 1 as the coverage requirement got stricter, overall coverage got reduced. For point 2 as the cost requirement was relaxed keeping coverage requirement as $\frac{1}{8}$ of M_ℓ, the overall coverage didn't increase. Refer to Table 2.12 for more details.

α	δ	Coverage
2	1/4	178
2	1/2	150
2	1/3	178
3	1/8	184
4	1/8	184
5	1/8	184
3	1/4	176

Table 2.12: Theoretical proofs give a (1/8, 2) approximation guarantee. This Table shows the tradeoffs between α and δ for Philosophers dataset. The values of α were varied keeping δ fixed and vice versa.

Although the above 2 variations also provided good coverage for lower budgets. Round provided better coverage consistently across all budget (cost) values. The scaling was used only for empirical analysis.

Scalability Study: In this section, we will analyze the scalability of Round.

Experimental Setup: For scalability study we used synthetic datasets (refer from 2.3) to analyse the scalability of Round.

Dataset	Round	Deterministic	Round (no scaling)	B
Genome (Threat)	216	220	234	5
Genome (Threat)	217	225	246	10
Genome (Threat)	192	207	248	15
Genome (Threat)	232	199	248	20
Genome (Threat)	214	235	248	25
Flickr	NA	NA	NA	5
Flickr	2165	1920	1911	10
Flickr	2178	2291	2170	15
Flickr	2235	2318	2200	20
Flickr	2095	1759	2195	25

Table 2.13: Coverage per rounding scheme for various datasets. 90% fixed coverage. The cost parameter varies from 5, 10, 15, 20, 25.

Observations: We observed that Round is quite scalable. The running time is dominated by the time needed to solve the LP. We use Gurobi solver, which is able to run successfully on datasets whose data matrix (i.e., the matrix of objects and tags) has up to 10^8 entries. In contrast, the ILP does not scale beyond datasets with more than 10^6 entries.

Dataset	$\|S\|$	$\|T\|$	ILP	Round
Genome (Threat)	248	4632	00:02:22	00:01:36
Uniref90	21537	2193	08:56:49	07:30:33
Synthetic 1 ($p = 0.05$)	100	100	00:01:30	00:01:10
Synthetic 2 ($p = 0.2$)	100	100	00:00:48	00:00:40
Synthetic 3 ($p = 0.05$)	1000	1000	00:02:05	00:01:47
Synthetic 4 ($p = 0.1$)	1000	1000	00:02:17	00:01:42
Synthetic 5 ($p = 0.15$)	1000	1000	00:02:72	00:02:16
Synthetic 6 ($p = 0.2$)	1000	1000	00:04:25	00:03:05
Synthetic 7 ($p = 0.05$)	10000	10000	NA	02:39:05
Synthetic 8 ($p = 0.05$)	1000	10000	NA	00:25:32

Table 2.14: ILP & Round run-times for various datasets. The Table shows that ILP did not scale beyond datasets with more than 10^6 entries. LP on the other hand ran successfully for datasets whose data matrices have upto 10^8 entries.

From Table 2.14 we can observe that ILP ran successfully for Uniref90 dataset. However, it did not run for Synthetic 7 dataset. The reason being the data matrix for Uniref90 is comparatively denser than the data matrix for Synthetic 7 and a single tag in Uniref90 was able to explain many rows as compared to that of Synthetic 7 dataset. To verify this scenario we created synthetic datasets (similar to those of Uniref90) and observed that ILP ran successfully.

2.5.5 Explanation of Clusters

Threat Dataset

Our method chose 13 tags for the harmful cluster. Upon expert review of our results, we found that certain tags served as indicators that genes found within the harmful cluster can intrinsically be viewed as harmful, while others may need to act in concert, be viewed in combination with other tags, or be representative of selection bias. Of the 13 tags selected, 4 indicate an intrinsic capability of being harmful: KW-0800 (toxin), 155864.Z3344 (Shiga toxin 1), IPR011050 (Pectin lyase fold/virulence), and IPR015217 (invasin domain). Another 4 tags are suggestive that the genes implicated are involved in processes or locations commonly associated with the threat: KW-0732 (signal peptide), KW-0614 (plasmid), KW-0964 (secreted), and GO:0050896 (response to stimulus). Other tags associated with the threat partition such as KW-0002 (3-D structure) indicate a limited amount of data and perhaps bias in the research literature for the clusters analyzed. Table 2.15 provides definitions of some of these tags.

2.6 Results Summary

This section gives a high level overview of the research questions and answers obtained.

The table below describes various rounding schemes implemented. These rounding schemes were implemented to understand and present a comparative study of various rounding

String	Keyword	Definition
KW-0800	Toxin	"Naturally-produced poisonous protein that damages or kills other cells, or the producing cells themselves in some cases in bacteria. Toxins are produced by venomous and poisonous animals, some plants, some fungi, and some pathogenic bacteria. Animal toxins (mostly from snakes, scorpions, spiders, sea anemones and cone snails) are generally secreted in the venom of the animal".
GO:0050896	response to stimulus	"Any process that results in a change in state or activity of a cell or an organism (in terms of movement, secretion, enzyme production, gene expression, etc.) as a result of a stimulus. The process begins with detection of the stimulus and ends with a change in state or activity or the cell or organism".
KW-0964	secreted	"Protein secreted into the cell surroundings".
GO:0050794	regulation of cellular process	"Any process that modulates the frequency, rate or extent of a cellular process, any of those that are carried out at the cellular level, but are not necessarily restricted to a single cell. For example, cell communication occurs among more than one cell but occurs at the cellular level".
GO:0016787	hydrolase activity	"Catalysis of the hydrolysis of various bonds, e.g. C-O, C-N, C-C, phosphoric anhydride bonds, etc. Hydrolase is the systematic name for any enzyme of EC class 3".
IPR011050	Pectin_lyase-fold/virulence	"Microbial pectin and pectate lyases are virulence factors that degrade the pectic components of the plant cell wall".
IPR015217	Invasin_dom_3	"It forms part of the extracellular region of the protein, which can be expressed as a soluble protein (Inv497) that binds integrins and promotes subsequent uptake by cells when attached to bacteria".
KW-0732	Signal	"Protein which has a signal sequence, a peptide usually present at the N-terminus of proteins and which is destined to be either secreted or part of membrane components".
GO:0034248	regulation of cellular amide metabolic process	"Any process that modulates the frequency, rate or extent of the chemical reactions and pathways involving amides".
KW-0614	Plasmid	"Protein encoded on a plasmid, a self-replicating circular DNA that is found in a variety of bacterial, archaeal, fungal, algal and plant species, and can be transferred from one organism to another. Plasmids often carry antibiotic-resistant genes and are widely used in molecular biology as vectors of genes and in cloning".
KW-1015	Disulfide bond	"Protein which is modified by the formation of a bond between the thiol groups of two peptidyl-cysteine residues".
KW-0002	3-D structure (KW-0002)	"P+A1:C14 or part of a protein, whose three-dimensional structure has been resolved experimentally (for example by X-ray crystallography or NMR spectroscopy) and whose coordinates are available in the PDB database".

Table 2.15: Tags selected by our algorithm for the harmful cluster in the Threat dataset. Red is an intrinsic threat. Blue is suggestive. Black is due to a lack of sufficient background.

schemes; and analyze the performance Round. This implementation was done as an extension to the performance research question.

S.No	Research Question	Results Summary
1	Effects of Parameters (Cost and Coverage) on solution quality and feasibility	An increase in cost increased the solution feasibility; decrease in coverage requirement increased the solution quality for datasets with sparse matrices; Round overcame the unbalanced coverage issue algorithm stated by Davidson et al., [DGR18]. The experiment were run for $k = 2$ (base case) and coverage requirement varied from 90%, 80%, 70%, 60% .
2	Pair of tags	The pair of tags when used as an additional attribute set increased the solution feasibility for a smaller budget. A qualitative analysis of the results was also done and can be referred to from the section above. The experiments were run for $k = 2$ (base case).
3	k-cluster Analysis	Here we focused on analyzing results when $k = 2$ vs $k = 4$ for same dataset. Due to lack of high density dataset, this experiment was run on Threat dataset. We observed that different set of attributes got picked when $k = 2$ vs $k = 4$. We did not observe an increase in solution feasibility or quality.
4	Performance	Round gave solutions close to the optimal solution; the approximation factors were always more than 0.8 (much higher than theoretical guarantee); Round is quite scalable, LP ran successfully for datasets whose data matrix has up to 10^8 entries. The experiments were run for $k = 2$.
5	Explanation of clusters	This sections provided a qualitative analysis of our findings. Our method chose 13 tags. Upon expert review of our results, we found certain tags served as indicators that genes found within the harmful cluster can intrinsically be viewed as harmful.

Table 2.16: The table gives a high level overview of the research questions and the results obtained. The details of the results can be referred to from the previous section.

Rounding scheme	Definition	Result Summary
Round	Algorithm as defined in Section 1.	Round gave high overall and cluster wise coverage.
Round: No scaling down	Fractional solution was not scaled down by a factor of 2.	This rounding scheme also provided high coverage as cost was increased. For the Threat dataset, as cost was increased the approximate ratio became equal to the optimal solution.
Round: Deterministic	Deterministic rounding of fractional values i.e., values ≥ 0.5 were rounded to 1 and values < 0.5 were rounded to 0	The rounding scheme did not always provided high coverage.

Table 2.17: High level summary of various rounding schemes and the results obtained. These results are an extension of the Performance section of the research questions described above. All experiments were run for $k = 2$ (base case). The Threat and Flickr datasets were considered for these experiments. Cost parameter varied from 5, 10, 15, 20 and 25.

Chapter 3

Related Work

The topic of “Explainable AI” [Gun17] has recently attracted a lot of attention especially in the context of supervised learning. In particular, many researchers have studied the topic in conjunction with methods in deep learning [Pro17, Pro18, DBH18, Mil18, MIN17, ZMLC18, ZC18]. To our knowledge, not much work has been done in the context of interpreting results from clustering. In [KRV+17], the authors consider the use of human judgement to interpret a given clustering as well as providing suggestions for improving the results. Their goal was to improve the clustering quality through human guidance and they used constraint programming techniques to obtain improvements. Other methods for improving a given clustering were considered in [DB10, QD09]. The notion of “descriptive clustering” studied in [TBHDKCV] is different from our work; their idea is to allow the clustering algorithm to use both the features of the objects to be clustered and the descriptive information for each object. They present methods that for constructing the Pareto frontier based on two objectives, one based on features and the other based on the descriptive information. Like [DGR18], the focus of our work is not on generating a clustering; instead, the goal is to explain the results of clustering algorithms.

Since there is not much literature available which is in line with our approach, we have studied various clustering techniques to get an understanding of the methods which are being used by the researchers and how different they are from our approach. Below are the clustering methods and techniques that are being used currently:

Predictive clustering: A technique [ŽDS05, SRW19, ZMZ$^+$19] of performing classification which finds clusters in the input attributes and homogeneity in the class labels at the same time. Earlier work [Langley, 1996] [Lan96] viewed decision trees as a predictive clustering where each leaf is a "cluster" with a homogeneous class label and some attributes (those on its path). More recent work by [Zenko et al., 2005] proposed learning predictive clustering rules.

Conceptual clustering: This technique [Jo19, RBK19] focuses on using a set of features to create the clusters and then uses the same set of features to explain the generated clusters. Gennari et al., 1989; Fisher, 1987 [GLF89, Fis87] tries to put objects into classes where each class is defined by a concept expressed in a given description language. The same set of features are used to form and describe the clusters. More work is being done in the field of conceptual clustering with constrained programming [MK10], however, these approaches again focus on explaining the clusters while generating them.

Data Clustering: A review by [JMF99, ZLM15] gives an overview of pattern clustering methods from a statistical pattern recognition perspective, with a goal of providing useful advice and references to fundamental concepts accessible to the broad community of clustering practitioners. According to Jain et al., 1999 following are the ways to analyze and cluster datasets:

Partitional clustering: A technique [XW05, HNE19, MAG19] to directly decompose the dataset into a set of disjoint clusters. More specifically, they attempt to determine an integer number of partitions that optimise a certain criterion function. The criterion function may emphasize the local or global structure of the data and its optimization is an iterative procedure. Hierarchical clustering proceeds successively by either merging smaller clusters into larger ones, or by splitting larger clusters. The result of the algorithm is a tree of clusters, called dendrogram, which shows how the clusters are related. By cutting the dendrogram at a desired level, a clustering of the data items into disjoint groups is obtained.

Density-based clustering: The key idea of this type of clustering is to group neighbouring objects of a data set into clusters based on density conditions [KKSZ11].

Grid-based clustering: This type of algorithms is mainly proposed for spatial data mining. Their main characteristic is that they quantize the space into a finite number of cells and then they do all operations on the quantized space [BJS19, BAL+19, CWB18].

There are many different algorithms [XT15] for finding clusters for the above mentioned categories. Thus, the datasets can be categorized into the following groups.

Statistical: which are based on statistical analysis concepts. They use similarity measures to partition objects and they are limited to numeric data.

Conceptual: which are used to cluster categorical data. They cluster objects according to the concepts they carry. Another classification criterion is the way clustering handles uncertainty in terms of cluster overlapping.

Fuzzy clustering: which uses fuzzy techniques to cluster data and they consider that an object can be classified to more than one clusters. This type of algorithms leads to clustering schemes that are compatible with everyday life experience as they handle the uncertainty of real data. The most important fuzzy clustering algorithm is Fuzzy C-Means (Bezdeck et al., 1984) [BEF84, T+16].

Crisp clustering, considers non-overlapping partitions meaning that a data point either belongs to a class or not. Most of the clustering algorithms result in crisp clusters, and thus can be categorized in crisp clustering.

All of these techniques either focus on explaining the clusters while generating them or generating the clusters. Our objective is very different from all of the above approaches.

We aim to explain the clusters after clustering is done without knowing about the technique that was used to generate the clusters. Moreover, we aim to use the attributes which were not used in the clustering technique to explain the results of the clustering.

Chapter 4

Conclusions

We evaluated the formulation proposed by Davidson et al., [DGR18] and Sambaturu et al., [SGD+19]. Our results show that Round performed very well in practice. Although theoretical results guarantee a coverage factor of 1/8, the empirical results show that Round performs much better and in most cases the performance guarantee is between 0.8 and 0.9. This suggests that Round gives solutions which are very close to the optimal. Round also scaled well for datasets whose data matrix had up to 10^8 entries. However, the ILP did not scale beyond datasets with more than 10^6 entries. To further empirically analyze Round, we implemented different rounding schemes and compared the results. We observed that Round performed better than the others. Although the other approaches gave very high coverage at lower costs, the coverage decreased as the cost increased.

Using different parameters such as coverage level, cost, and overlap, we obtained a range of solutions from which a practitioner can choose appropriate descriptors. Computational experiments suggest that Round performs much better in terms of per cluster coverage compared to the approach proposed by Davidson et al., [DGR18]. For instances, where the solution was infeasible for a lower cost, we filled the gap using "Bounded Overlap" and "Pair of tags" versions of MinConCD. It was observed that implementing these versions provided feasibility when the solution was infeasible. By allowing a minimum overlap of descriptors between the 2 clusters, we could ensure a feasible solution by still minimizing the number of descriptors selected and maximizing the number of objects covered.

Although Round has good performance in most cases, it has a few limitations as well. The algorithm does not perform as expected for datasets with an extremely sparse data matrix (a dataset where each object has very few descriptors associated with it). It also does not perform as expected in cases where coverage is not a constant factor of the number of objects in each cluster.

4.1 Future Work

This section talks about the future work and extensions of MinConCD and Round. Following are a few suggestions:

- What will be the impact on solution quality if a pair of tags is charged two instead of one?

  In the previous Section (add reference) we observed that pair of tags picked did provide a better explanation of the clusters without increasing the overall cost. However, in these results every pair of tag is charged as one. The next steps can be to understand the impact on solution quality if a pair of tag is charged two instead of one.

- How will Round perform if coverage requirement is arbitrary?

  So far we have considered the coverage requirement to be a constant factor. The next steps can be to understand the performance of Round if coverahe requirement is arbitrary for some or all of the clusters.

Bibliography

[BAL$^+$19] Thapana Boonchoo, Xiang Ao, Yang Liu, Weizhong Zhao, Fuzhen Zhuang, and Qing He. Grid-based dbscan: Indexing and inference. *Pattern Recognition*, 90:271–284, 2019.

[BEF84] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

[BGLL08] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[BJS19] Daniel Brown, Arialdis Japa, and Yong Shi. A fast density-grid based clustering method. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0048–0054. IEEE, 2019.

[Bol13] Marianna Bolla. *Spectral clustering and biclustering: Learning large graphs and contingency tables.* John Wiley & Sons, 2013.

[CWB18] Wei Cheng, Wei Wang, and Sandra Batista. Grid-based clustering. In *Data Clustering*, pages 128–148. Chapman and Hall/CRC, 2018.

[DB10] Xuan Hong Dang and James Bailey. Generation of alternative clusterings using the cami approach. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 118–129. SIAM, 2010.

[DBH18] Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International convention on information* *and communication technology, electronics and microelectronics (MIPRO)*, pages 0210–0215. IEEE, 2018.

[DGR18] Ian Davidson, Antoine Gourru, and S Ravi. The cluster description problem-complexity results, formulations and approximations. In *Advances in Neural Information Processing Systems*, pages 6190–6200, 2018.

[Fis87] Douglas H Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2(2):139–172, 1987.

[For10] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[GF16] Bryce Goodman and Seth Flaxman. Eu regulations on algorithmic decision-making and a "right to explanation". In *ICML workshop on human interpretability in machine learning (WHI 2016), New York, NY. http://arxiv.org/abs/1606.08813 v1*, 2016.

[GGU72] Michael R Garey, Ronald L Graham, and Jeffrey D Ullman. Worst-case analysis of memory allocation algorithms. In *Proceedings of the fourth annual ACM symposium on Theory of computing*, pages 143–150. ACM, 1972.

[GLF89] John H Gennari, Pat Langley, and Doug Fisher. Models of incremental concept formation. *Artificial intelligence*, 40(1-3):11–61, 1989.

[Gun17] David Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2017.

[HF18] Md Nafiz Hamid and Iddo Friedberg. Identifying antimicrobial peptides using word embedding with deep recurrent neural networks. *BioRxiv*, page 255505, 2018.

[HNE19] Mohamed Aymen Ben HajKacem, Chiheb-Eddine Ben N'Cir, and Nadia Essoussi. Overview of scalable partitional methods for big data clustering. In *Clustering Methods for Big Data Analytics*, pages 1–23. Springer, 2019.

[HPK11] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[HSSC08] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[JGK18] Aashish Jain, Hareesh Gali, and Daisuke Kihara. Identification of moonlighting proteins in genomes using text mining techniques. *Proteomics*, 18(21-22):1800083, 2018.

[JK18] Aashish Jain and Daisuke Kihara. Phylo-pfp: improved automated protein function prediction using phylogenetic distance of distantly related sequences. *Bioinformatics*, 35(5):753–759, 2018.

[JMF99] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[Jo19] Taeho Jo. Text clustering: Conceptual view. In *Text Mining*, pages 183–201. Springer, 2019.

[Joh74] David S Johnson. Approximation algorithms for combinatorial problems. *Journal of computer and system sciences*, 9(3):256–278, 1974.

[KKSZ11] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011.

[KRV$^+$17] Chia-Tung Kuo, SS Ravi, Christel Vrain, Ian Davidson, et al. A framework for minimal clustering modification via constraint programming. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[KT06] J. Kleinberg and E. Tardos. *Algorithm Design*. Pearson Publishing, New York, NY, 2006.

[Lan96] Pat Langley. *Elements of machine learning*. Morgan Kaufmann, 1996.

[MAG19] Mahesh Motwani, Neeti Arora, and Amit Gupta. A study on initial centroids selection for partitional clustering algorithms. In *Software Engineering*, pages 211–220. Springer, 2019.

[Mil18] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2018.

[MIN17] YAO MING. A survey on visualization for explainable classifiers. 2017.

[MK10] Marianne Mueller and Stefan Kramer. Integer linear programming models for constrained clustering. In *International Conference on Discovery Science*, pages 159–173. Springer, 2010.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Cambridge university press, 1995.

[MRS$^+$98] Madhav V Marathe, Ramamoorthi Ravi, Ravi Sundaram, SS Ravi, Daniel J Rosenkrantz, and Harry B Hunt III. Bicriteria network design problems. *Journal of algorithms*, 28(1):142–171, 1998.

[Pro17] Proceedings of the IJCAI-17 Workshop on Explainable AI (XAI). Available from: http://www.intelligentrobots.org/files/IJCAI2017/IJCAI-17_XAI_WS_Proceedings.pdf, 2017.

[Pro18] Proceedings of the IJCAI-ECAI-2018 Workshop on Explainable AI (XAI). Available from https://www.dropbox.com/s/jgzkfws41ulkzxl/proceedings.pdf?dl=0, 2018.

[QD09] ZiJie Qi and Ian Davidson. A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 717–726. ACM, 2009.

[RBK19] Enrique H Ruspini, James C Bezdek, and James M Keller. Fuzzy clustering: A historical perspective. *IEEE Computational Intelligence Magazine*, 14(1):45–55, 2019.

[RW18] Shreyas Ramesh and Andrew Warren. The learnability of taxonomic divisions. Talk and poster presentation at ISMB, 2018.

[SGD$^+$19] Prathyush Sambaturu, Aparna Gupta, Ian Davidson, S S Ravi, Anil Vullikanti, and Andrew Warren. Generating near-optimal cluster descriptors for explainability. *(Submitted)European Conference on Machine Learning and Principles and Practice of Knowledge Discovery*, 2019.

[SRW19] Jaehyeok Shin, Alessandro Rinaldo, and Larry Wasserman. Predictive clustering. *arXiv preprint arXiv:1903.08125*, 2019.

[T$^+$16] Pham Huy Thong et al. A novel automatic picture fuzzy clustering method based on particle swarm optimization and picture composite cardinality. *Knowledge-Based Systems*, 109:48–60, 2016.

[Tan18] Pang-Ning Tan. *Introduction to data mining*. Pearson Education India, 2018.

[TBHDKCV] Chia-Tung Thi-Bich-Hanh Dao, SS Kuo, and Ian Davidson Christel Vrain. Descriptive clustering: Ilp and cp formulations with applications.

[Udw18] Rajan Udwani. Multi-objective maximization of monotone submodular functions with cardinality constraint. In *Advances in Neural Information Processing Systems*, pages 9493–9504, 2018.

[WS11] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.

[XT15] Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.

[XW05] Rui Xu and Donald C Wunsch. Survey of clustering algorithms. 2005.

[YML13] Jaewon Yang, Julian McAuley, and Jure Leskovec. Community detection in networks with node attributes. In *2013 IEEE 13th International Conference on Data Mining*, pages 1151–1156. IEEE, 2013.

[ZC18] Yongfeng Zhang and Xu Chen. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*, 2018.

[ŽDS05] Bernard Ženko, Sašo Džeroski, and Jan Struyf. Learning predictive clustering rules. In *International Workshop on Knowledge Discovery in Inductive Databases*, pages 234–250. Springer, 2005.

[ZLM15] Btissam Zerhari, Ayoub Ait Lahcen, and Salma Mouline. Big data clustering: Algorithms and challenges. In *Proc. of Int. Conf. on Big Data, Cloud and Applications (BDCA'15)*, 2015.

[ZMJM14] Mohammed J Zaki, Wagner Meira Jr, and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[ZMLC18] Zhiwei Zeng, Chunyan Miao, Cyril Leung, and Jing Jih Chin. Building more explainable artificial intelligence with argumentation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[ZMZ$^+$19] Daiane Aparecida Zuanetti, Peter Müller, Yitan Zhu, Shengjie Yang, and Yuan Ji. Bayesian nonparametric clustering for large data sets. *Statistics and Computing*, 29(2):203–215, 2019.