Combinatorial Feature Selection
Parameterized Algorithms and Complexity

Vincent Froese

Masterarbeit

Zur Erlangung des akademischen Grades
Master of Science (M.Sc.)
im Studiengang Informatik

Technische Universität Berlin
Fakultät IV - Elektrotechnik und Informatik
Institut für Softwaretechnik und Theoretische Informatik
Fachgebiet Algorithmik und Komplexitätstheorie

Eingereicht von Vincent Froese

Betreuer: Dipl.-Inf. René van Bevern,
Prof. Dr. Rolf Niedermeier,
Dipl.-Inf. Manuel Sorge

28. September 2012

Eidesstattliche Erklärung

Die selbständige und eigenhändige Ausfertigung versichert an Eides statt

Berlin, den 28. September 2012

.....
Unterschrift

Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit Problemen aus dem Bereich der Kombinatorischen Merkmalsselektion. Ziel der Kombinatorischen Merkmalsselektion ist es, für eine gegebene Menge hochdimensionaler Daten eine Teilmenge der Dimensionen so auszuwählen, dass die Daten in den ausgewählten Dimensionen eine gesuchte Eigenschaft erfüllen. Probleme dieser Art treten zum Beispiel im Bereich des Maschinellen Lernens auf, wo Daten häufig erst geeignet für die jeweilige Zielanwendung vorverarbeitet werden müssen. In dieser Arbeit betrachten wir Probleme aus einem allgemeinen Framework für Kombinatorischen Merkmalsselektion, das von Charikar et al. [CGK+00] eingeführt wurde. Dieses Framework umfasst unter anderem das Problem Hidden Clusters aus dem Bereich der Clusteranalyse, sowie das Dimensionsreduktionsproblem Distinct Vectors. Bei Hidden Clusters möchte man verrauschte Dimensionen löschen, sodass sich die Daten in den übrigen Dimensionen in eine vorgegebene Anzahl an Clustern einteilen lassen. Das Distinct Vectors Problem besteht darin, eine möglichst kleine Menge an Dimensionen zu behalten, die ausreicht, um alle Datenpunkte unterscheiden zu können. Außerdem führen wir ein weiteres Clustering Problem namens Hidden Cluster Graph ein, bei dem die Anzahl an Clustern nicht vorgegeben ist.

Diese Arbeit beinhaltet eine verfeinerte Analyse hinsichtlich der Komplexität der oben genannten Probleme. Dabei setzen wir die Ergebnisse von Charikar et al. [CGK+00] fort und untersuchen die Probleme im Kontext der Parametrisierten Komplexitätstheorie. Zu den erzielten Ergebnissen gehören unter anderem parametrisierte Härtebeweise bezüglich der Parameter Anzahl auszuwählender Dimensionen und Anzahl zu löschender Dimensionen. Darüber hinaus werden einige Spezialfälle der genannten Probleme formuliert und analysiert. Diese Spezialfälle konzentrieren sich auf weitere Parameter, wie die gesuchte Anzahl an Clustern, der Clusterradius, die Alphabetgröße oder den paarweisen Hamming Abstand der Daten. Zwar gelten einige der oben erwähnten Härteresultate für manche Probleme sogar in den eingeschränkten Spezialfällen, doch ist es dennoch möglich, für geeignete Parameterkombinationen Lösungsalgorithmen anzugeben, bei denen der superpolynomielle Teil der Laufzeit sich auf den Parameter beschränkt. Für Distinct Vectors zeigen wir zudem eine untere Schranke an den superpolynomiellen Teil der Laufzeit für den kombinierten Parameter Anzahl auszuwählender Dimensionen und Alphabetgröße. Die Ergebnisse geben Anlass für zukünftige Forschung auf dem Gebiet der Kombinatorischen Merkmalsselektion.

Abstract

This work deals with the topic of combinatorial feature selection. Given a set of high-dimensional data, the goal is to select an appropriate subset of dimensions such that some desired property holds for the data restricted to the selected dimensions. Problems of this kind arise as data preprocessing tasks in areas such as machine learning. Charikar et al. [CGK+00] defined a general framework for combinatorial feature selection. Their framework comprises cluster analysis as well as dimension reduction. They defined several instances of feature selection problems within their framework and provided hardness results as well as approximation algorithms for them. In this work, we consider two of their problems: The clustering problem Hidden Clusters and the dimension reduction problem Distinct Vectors. The goal of Hidden Clusters is to get rid of noisy dimensions in the data such that in the remaining dimensions the data can be divided into a given number of clusters. The Distinct Vectors problem aims at finding a minimum number of dimensions such that all given points can still be distinguished from each other in the selected dimensions. In addition to the two problems mentioned above, we introduce another clustering problem, called Hidden Cluster Graph, where the number of cluster centers is not known.

We conduct a refined analysis concerning the computational complexity of the above problems from the perspective of parameterized complexity analysis. In doing so, we pursue the analysis of Charikar et al. [CGK+00] and provide parameterized hardness results as well as fixed-parameter algorithms. For all three problems, it turns out that they are hard to solve with respect to some natural parameters such as the number of dimensions to select or the number of dimensions to delete. In order to obtain fixed-parameter tractability, we also focus on some special cases of the problems involving parameters such as the number of cluster centers, the radius of a cluster, the size of the alphabet, or the pairwise Hamming distance of the data points. We show that the problems are indeed fixed-parameter tractable for several combinations of the mentioned parameters by providing problem kernels as well as fixed-parameter tractable algorithms. Moreover, we prove a lower bound on the running time for any fixed-parameter algorithm for Distinct Vectors parameterized by the number of dimensions to select and the size of the alphabet. We also indicate some interesting open questions resulting from our discussions which encourage for future research.

Contents

1 Introduction

Imagine a professional whisky taster (presumably a Scotsman) whose job is to taste whiskies and afterwards judge them by their quality. In order to do a good job, he always takes notes of each whisky he tastes. Carefully, he writes down all the information that help him in assessing a particular whisky such as color, age, bouqet, taste, alcohol strength, viscosity, the type of cask used for maturation, the "peatiness", and many attributes more. After many years and thousands of glasses of whisky, he decides to share his knowledge and experience with other enthusiasts by writing a comprehensive book about Scotch whisky containing a detailed description of every single whisky he knows of. His bulk of notes, however, is way too much to be completely contained in his book. Hence, he has to find a compact description containing only those attributes that are necessary to uniquely describe an individual whisky. Moreover, he wants to categorize the whiskies according to their character. There are five to six geographical regions in Scotland that are recognized to produce whiskies of distinct types [Hof07]. His goal is thus to divide up the whiskies into five groups such that all whiskies of one group are as similar to each other in as many attributes as possible.

The two problems of the whisky connoisseur described above can in fact be considered as instances of what we call *combinatorial feature selection* problems. The term combinatorial feature selection refers to a general class of problems that, given a set of high-dimensional objects, ask for selecting a subset of dimensions (*features*) such that some desired property holds for the dataset restricted to the selected dimensions. Problems of this kind often arise in areas like machine learning [HM94, BL97, WMC$^+$00], data mining [LM07, LM98a, LM98b] or computational biology [GWBV02], where the dimensionality of the considered data frequently is in the thousands. Examples of applications involving large feature sets are document classification [FH01] or gene expression array analysis [XK01]. For many applications, large feature sets impose severe problems regarding efficiency and computational tractability of data processing as well as the accuracy of the results (a phenomenon often referred to as the *curse of dimensionality* [Bel61]). In most cases an appropriate preprocessing of the data is inevitable, and feature selection is one common approach to it. Working only on a carefully chosen subset of features provides several beneficial effects such as increased tractability of data processing, better generalization performance, decreased risk of overfitting and elimination of noise in the data. Moreover, it can bring a better understanding of the structure underlying the data and it may enable visualizations.

The classical approach to feature selection takes place in an *affine* setting where we are allowed to select an affine subspace of the original feature space (in this context, the term *feature extraction* is often used since new features are constructed out of the original ones). Often the resulting features take the form of linear combinations of the

original features. Affine versions of feature selection, like principal component analysis [Pea01, Jol02] for example, are well-studied in the literature [Krz87]. In combinatorial feature selection [CGK+00], we directly choose a subset of the original dimensions. One advantage of the combinatorial approach is that the selected subspace is interpretable in the sense that the selected features have a clear meaning in the context of the original data. Moreover, applying an already determined solution to new datasets can be done much faster in a combinatorial setup since it only discards features, whereas an affine method often involves some sort of linear transformation. Another advantage is that combinatorial feature selection can be applied to arbitrary feature spaces containing categorical or symbolic attributes, whereas affine methods often require numerical feature spaces (real-valued vector spaces).

In section 1.1 we describe a general theoretical framework that was introduced by Charikar et al. [CGK+00] for studying combinatorial feature selection problems. They considered several instances of combinatorial feature selection problems within this framework and provided approximation algorithms as well as hardness results. In this work we investigate some of their problems from the perspective of parameterized complexity analysis [DF99, FG06, Nie06]. section 1.2 gives an overview of this work and the accomplished results.

1.1 Combinatorial Feature Selection: A Framework

This section contains the basic definitions of combinatorial feature selection problems which we consider throughout this work. We mainly adopt the notation by Charikar et al. [CGK+00, Section 2].

In the following, $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ denotes a set of n points[1] of a d-dimensional feature space Σ^d. We refer to a specific dimension by using its index $i \in \{1, \ldots, d\}$. Accordingly, $(x)_i$ denotes the value of x in dimension i. Let $K \subseteq \{1, \ldots, d\}$ be a subset of dimensions. Then $x_{|K} \in \Sigma^{|K|}$ is the projection of x onto the subspace indexed by the dimensions in K and we define the set $S_{|K} := \{x_{|K} \mid x \in S\}$. The feature space Σ^d does not necessarily have to be a metric space, we only assume that it is equipped with a function $\mathrm{dist} : \Sigma^d \times \Sigma^d \to \mathbb{Q}$, which defines a distance between points from Σ^d. Moreover, for any subset K of dimensions, the restriction $\mathrm{dist}_{|K} : \Sigma^{|K|} \times \Sigma^{|K|} \to \mathbb{Q}$ of the function dist to the subspace indexed by the dimensions in K has to be defined.

Feature selection is the task of selecting a subset of dimensions K such that $S_{|K}$ satisfies a given property Π. More precisely, we consider an optimization setting where we ask for a subset K of minimum or maximum cardinality such that Π holds for $S_{|K}$. Whether the goal is to minimize or to maximize the size of K depends on Π. Charikar et al. [CGK+00] defined two complementary flavors of the feature selection problem which they called *subspace selection* and *dimension reduction*.

[1] Depending on Σ, these may be vectors from a vector space over some field or simply words over an alphabet. We use the general term "points" for convenience as the meaning will be clear from the context.

Subspace Selection. In this genre of feature selection problems we are given a set S that does not satisfy a certain property Π and we want to find a subset of dimensions K of maximum cardinality such that $\Pi(S_{|K})$ holds. Problems of this kind may arise if Π is such that it reveals some interesting structure that "explains" the data in some way. The goal is then to maintain a maximum amount of information (features) from the input, subject to preserving the underlying structure, that is, we want to be able to explain (or understand) as much of the data as possible. For example, data clustering problems can be formulated in this way. We will consider the Hidden Clusters problem by Charikar et al. [CGK+00] in which we are given a set of points from a high-dimensional feature space which cannot be clustered due to the presence of some "noisy" dimensions. The goal is to discard the noisy dimensions such that in the remaining dimensions the data clusters well. They defined clustering in terms of a min-max objective where the task is to find a given number of cluster centers such that the maximum distance of a point to its corresponding cluster center is minimized. The formal problem definition reads as follows:

Hidden Clusters
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ consisting of n points in d dimensions, $r \in \mathbb{Q}$, $\ell, k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that there exist ℓ centers $C = \{c_1, \ldots, c_\ell\} \subseteq \Sigma^d$ and an assignment of points to centers $\sigma : S \to C$ such that $\text{dist}_{|K}(x, \sigma(x)) \leq r$ for all points $x \in S$?

Figure 1.1 shows an illustrative example of the Hidden Clusters problem.

In addition to Hidden Clusters, we introduce another clustering problem that belongs to the setting of subspace selection. One drawback of the Hidden Clusters formulation is the requirement to know the number of clusters ℓ beforehand. As this may often not be the case in practical applications, we introduce a somewhat stricter notion of clustering, called a *cluster graph*. Herein, a cluster is a set of points having distance of at most r from each other and a distance greater than r to all other points. The task can be described as minimizing the distances of points within a cluster (increasing homogeneity of a cluster) while maximizing the distances between clusters (increasing heterogeneity between clusters). In this way, the number of resulting clusters depends on the data (for a given radius r). We call this problem the Hidden Cluster Graph problem. An example instance is shown in Figure 1.2. The problem is defined as follows:

Hidden Cluster Graph
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ consisting of n points in d dimensions, $r \in \mathbb{Q}$, $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that the graph $G_K = (V, E_K)$ with

$$V := S, \quad E_K := \{\{x_i, x_j\} \mid x_i \neq x_j \in V,\ \text{dist}_{|K}(x_i, x_j) \leq r\}$$

is a cluster graph (that is, a union of disjoint cliques)?

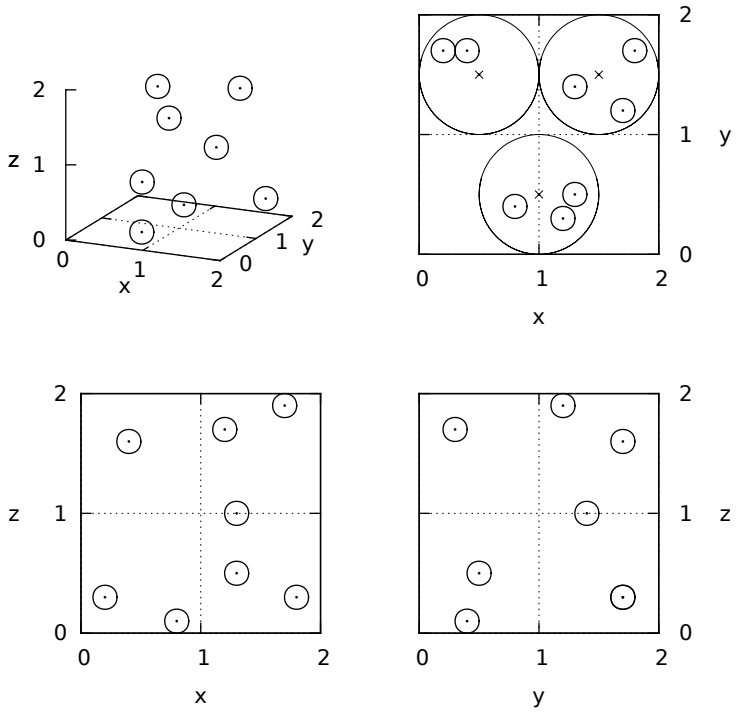

Figure 1.1: Example of the HIDDEN CLUSTERS problem: The top left plot shows a three-dimensional dataset which cannot be separated into three clusters of radius 0.5. The top right plot shows a solution to the problem where the dataset is projected onto the x-y-plane. In these two dimensions the data fulfills the clustering condition. The two bottom plots contain the projections onto the x-z-plane and the y-z-plane. In both cases the data does not fulfill the clustering condition.

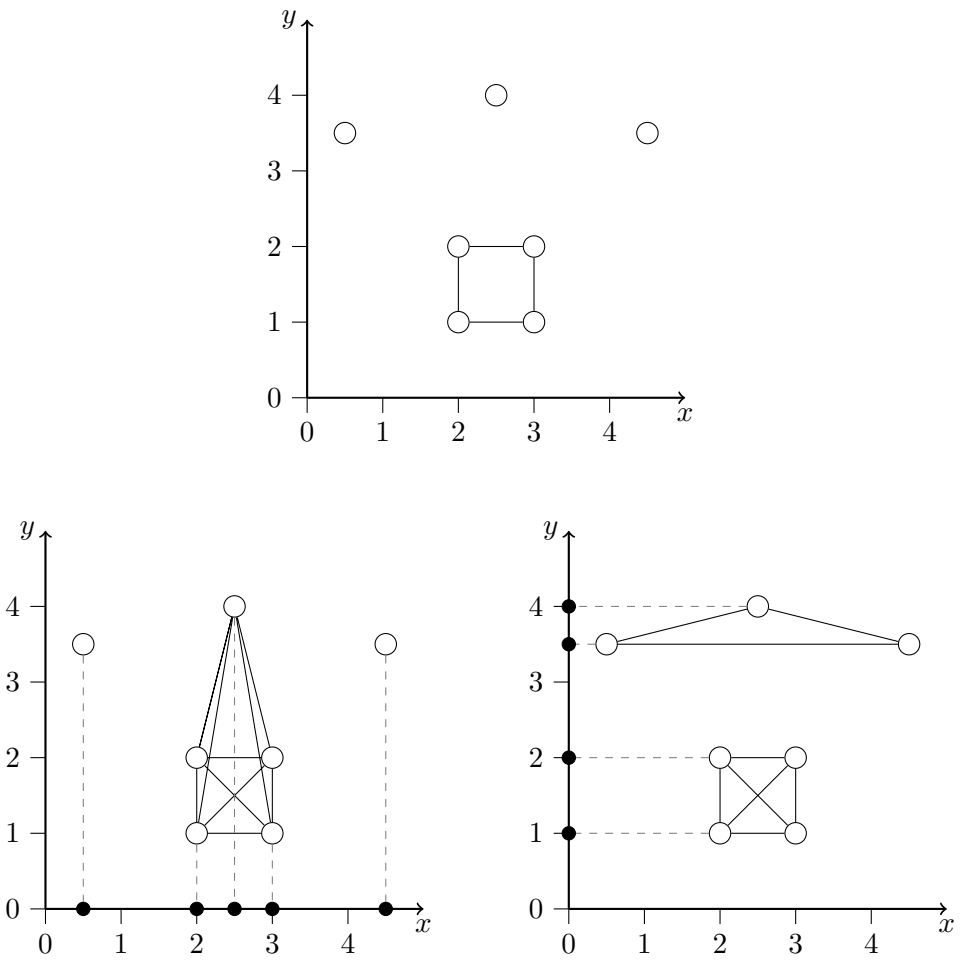

Figure 1.2: Example of the Hidden Cluster Graph problem: The top plot shows a two-dimensional dataset and the corresponding graph $G_{\{x,y\}}$, which is not a cluster graph with respect to the Euclidean metric and the radius $r = 1$. The two bottom plots illustrate the two possible solutions obtained by selecting either the dimension x (left) or y (right). The black dots indicate the projected data points. In both cases the corresponding graph is a cluster graph.

	1	2	3	4	5	6	7	8	9	10
x_1	0	1	1	1	1	0	1	1	0	0
x_2	0	0	0	1	1	0	1	1	0	0
x_3	0	1	0	0	0	0	1	1	1	1
x_4	1	0	1	1	0	0	1	0	1	0
x_5	1	1	0	1	1	0	0	0	1	0

Figure 1.3: Example of the Distinct Vectors problem: The dataset $\{x_1, \ldots, x_5\}$ consisting of five points over a binary alphabet in ten dimensions is represented as a matrix with rows corresponding to points and columns corresponding to dimensions. It is possible to distinguish all points from each other by selecting dimension 2, 5 and 7 (highlighted in gray).

Dimension Reduction. In this scenario we are given a set S satisfying a property Π and we want to find the smallest subset of dimensions K such that $\Pi(S_{|K})$ still holds. In contrast to subspace selection, we now want to determine the minimum amount of information (features) that is needed in order to appropriately describe the data. The goal of finding the smallest feature set that serves a given purpose may be motivated, for example, by aiming at a better understanding of the data or reducing the amount of resources involved in data processing by filtering out unnecessary or redundant information.

Charikar et al. [CGK+00] provided several instances of dimension reduction problems. In this work we consider one of their problems called Distinct Vectors[2]. This problem consists of the basic task to find a smallest subset of dimensions that suffices to distinguish all points in a given dataset. In its general formulation, the problem may arise in applications such as finding a unique key in a database or simply compressing data without losing the essential information to tell apart all data points. The formal decision problem reads as follows:

Distinct Vectors
Input: A multiset $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ of n distinct points in d dimensions and $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \leq k$ such that all points in $S_{|K}$ are still distinct?

Note that, for technical reasons, in the definition of Distinct Vectors the datasets S and $S_{|K}$ are defined to be multisets, that is they are allowed to contain multiple elements. This is required for the Distinct Vectors problem in order to be well-defined because otherwise all elements of a set $S_{|K}$ would be distinct by definition. Figure 1.3 depicts an example of a Distinct Vectors instance.

[2]Note that the term "vectors" is not used in the mathematical sense of an element of some vector space, but in the general sense of a word over an alphabet. Anyhow, we stick to the original problem name.

1.2 Overview and Results

In chapter 2 we give a brief introduction into the basics of parameterized complexity theory and graph theory.

chapter 3 and chapter 4 constitute the main part of this work. The main results obtained in these chapters are briefly summarized in Table 1.1. In chapter 3 we discuss the subspace selection problems Hidden Clusters and Hidden Cluster Graph. section 3.1 contains a detailed version of the proof sketch for NP-hardness of Hidden Clusters by Charikar et al. [CGK$^+$00]. The proof additionally yields W[1]-hardness with respect to the number of selected dimensions k implying that Hidden Clusters is unlikely to be solvable by an algorithm whose super-polynomial part of the running time only depends on k. The problem, however, is fixed-parameter tractable with respect to the dual parameter t, that is, the "number of dimensions to be deleted", combined with the number of centers ℓ: We give a fixed-parameter algorithm running in $O(\ell^t d(\ell^3 + n))$ time. Since we want to retain a maximum number of dimensions, the value of t might be small in applications. If the number of centers ℓ is small too, then the above algorithm solves the problem efficiently. Table 3.1 summarizes the results in more detail.

In section 3.2 we study the Hidden Cluster Graph problem equipped with distance functions induced by L_p-norms. We show NP-hardness as well as W[2]-hardness with respect to the parameter t for all natural numbers $p \geq 1$. On the contrary, for the L_∞-distance, we are able to solve the problem in $O(d(n^2 d + n^3)$ time. Hidden Cluster Graph becomes fixed-parameter tractable if the radius r is also taken into consideration as a parameter. This may be interesting for datasets that are normalized such that all points are located within a bounded region of the feature space (for example, inside a sphere of a small radius). The radius will then also be bounded. The combined parameter (r, t) allows an algorithm running in $O((2^p r)^t \cdot (n^2 d + n^3))$ time. See Table 3.2 for the details.

chapter 4 deals with the Distinct Vectors problem. We prove a dichotomy result concerning the special case of a binary alphabet: If, for each pair of points, the number of dimensions in which both have different entries (also called the Hamming distance) can be bounded from above by three, then Distinct Vectors can be solved in $O(n^3 d)$ time. Otherwise it is NP-hard. Furthermore, we analyze the parameterized complexity of Distinct Vectors. Despite the general NP-hardness, there may be hope for fixed-parameter tractability, for example with respect to the sought solution size k for this can be expected to take on small values. We show that, in general, this is not the case by proving W[2]-hardness with respect to k for an alphabet of unbounded size. It is therefore natural to add the alphabet size σ to the parameterization in order to obtain fixed-parameter tractability. Indeed, we prove existence of an $O(\sigma^{\sigma^k + k})$-size problem kernel. For constant alphabet size σ, Distinct Vectors is thus fixed-parameter tractable with respect to k. But the proof of NP-hardness also implies EW[2]-hardness for the combined parameter (k, σ). This basically means that Distinct Vectors is unlikely to be solvable by an algorithm whose running time depends singly exponential on k and σ. Note that for Distinct Vectors this implies that there is no linear-size

Table 1.1: Overview of the main results.

Problem	Results[†]
Hidden Clusters (See Table 3.1 for the details.)	NP-hard W[1]-hard with respect to k FPT with respect to (ℓ, t)
Hidden Cluster Graph (See Table 3.2 for the details.)	NP-hard W[2]-hard with respect to t FPT with respect to (r, t)
Distinct Vectors (See Table 4.1 for the details.)	NP-hard for $\sigma = 2$ and $f \geq 4$ polynomial-time solvable for $\sigma = 2$ and $f \leq 3$ W[2]-hard with respect to k W[1]-hard with respect to t for $\sigma = 2$ and $f \geq 4$ FPT with respect to (k, σ) and (k, f)

[†] k: sought solution size, t: number of dimensions to delete, ℓ: number of cluster centers, r: radius, σ alphabet size, f: maximum pairwise Hamming distance of the data points

problem kernel with respect to (k, σ) since this would allow to solve the problem in $2^{O(k+\sigma)} \cdot (nd)^{O(1)}$ time by trying out all subsets of dimensions.

For the dual parameter t, however, we show that the problem is W[1]-hard even for constant $\sigma = 2$. We define another parameter, called f, which denotes the bound on the maximal pairwise Hamming distance mentioned above. This parameter could be small in applications where the data is sparse (that is, most entries are 0). We develop some fixed-parameter tractability results for the combined parameter (k, f). A detailed overview of all results is given in Table 4.1.

1.3 Related Work

Feature selection is most commonly done within the affine setting. For example, principal component analysis, independent component analysis, canonical correlation analysis, or multidimensional scaling are well-studied methods for selecting affine linear subspaces. They are covered, for example, by the books of Duda et al. [DHS01] and Bishop [Bis06]. Nonlinear dimension reduction techniques can be found in the work by Schölkopf et al. [SSM98], Tenenbaum et al. [TSL00] or Roweis and Saul [RS00]. Further, see the surveys of Molina et al. [MBN02] and Guyon and Elisseeff [GE03] for a broad overview of different approaches to feature selection.

A popular problem of the combinatorial feature selection setup is Minimum Feature Set, which considers a given dataset of binary points that is divided into two different classes. The task is to choose a subset of at most k dimensions that allows to distinguish all pairs of points from different classes. The Distinct Vectors problem

by Charikar et al. [CGK+00] is a modification of the classic Minimum Feature Set problem in that it requires to distinguish all points from one another. Davies and Russel [DR94] proved Minimum Feature Set to be NP-complete. In addition, Van Horn and Martinez [HM94] showed that the problem is in fact hard to approximate in polynomial time via reduction from Set Cover. They proved that Minimum Feature Set cannot be approximated within a factor of $o(\log n)$ in polynomial time, unless NP $\subseteq$ DTIME$[n^{\log \log n}]$. Conversely, as Oliveira and Sangiovanni-Vincentelli [OSV92] observed, Minimum Feature Set can in turn be reduced to Set Cover. This allows for polynomial-time factor-$O(\log n)$ approximation algorithms, for example, see the work by Dash [Das97].

Parameterized complexity results for the Minimum Feature Set problem can be found in the work by Cotta and Moscato [CM03], where they proved W[2]-hardness with respect to the sought solution size k. This result is surprising insofar as one may reckon that it is in fact easier to distinguish only pairs of points from two different classes instead of all pairs. But, as we will see, Distinct Vectors is fixed-parameter tractable with respect to k for binary data, whereas Minimum Feature Set is W[2]-hard. Moreover, Cotta and Moscato recognized that even the special case of finding a feature set that distinguishes only a single point from all others (One-Out Feature Set) is NP-complete and W[2]-hard with respect to k [CM05]. They also identified an amenable variant called d-MaxRowWeight One-Out Feature Set, where the maximum number d of 1's for each point is bounded by a constant. They proved this case to be fixed-parameter tractable with respect to the parameter k [CM05].

2 Preliminaries

In this chapter we introduce the theoretical basis that is necessary to understand the subsequent chapters. We assume the reader to be familiar with the fundamentals in logic, set theory and computational complexity, especially with the concept of NP-completeness. For example, see the books by Papadmitriou [Pap94] and Arora and Barak [AB09] for comprehensive introductions. A detailed introduction to the concept of NP-completeness is given by Garey and Johnson [GJ79].

Miscellaneous and Notation. The cardinality of set A is denoted by $|A|$. A disjoint union of two sets A and B is denoted by $A \uplus B$. A *partition* of a set A is a division of A into disjoint, non-empty subsets. The Stirling number of the second kind

$$S(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^{k-i}\binom{k}{i}i^n$$

is the number of different partitions of an n-element set into exactly k non-empty subsets. By $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ we denote the sets of the natural, integer, rational and real numbers respectively. By $\vec{x}$ refer to the vector where all components are equal to x. We assume all numbers appearing as inputs in problem definitions and algorithms to be rational numbers. This is required for computational reasons since a Turing machine cannot handle irrational numbers in finite time. Moreover, throughout this work, we assume that *arithmetical operations* such as additions and comparisons of two numbers can be done in $O(1)$ time.

Norms and Metrics. For $p \in \mathbb{N}$ with $p \geq 1$, we define the L_p-norms

$$\|\cdot\|_p : \mathbb{R}^d \to [0,\infty), \quad \|x\|_p := \Big(\sum_{j=1}^{d} |(x)_j|^p\Big)^{\frac{1}{p}}$$

and for $p = \infty$ we define the *maximum norm* $\|x\|_\infty := \max_{j\in\{1,\dots,d\}} |(x)_j|$. One important property of a norm $\|\cdot\|$ is the so called *triangle inequality*

$$\forall x, y \in \mathbb{R}^d : \|x+y\| \leq \|x\| + \|y\|.$$

The L_p-norm induces the following *metric*

$$M_p : \mathbb{R}^d \times \mathbb{R}^d \to [0,\infty), \quad M_p(x,y) := \|x-y\|_p,$$

which defines a *distance* between two vectors. The *Hamming distance* $\Delta(x,y)$ of two words $x, y \in \Sigma^d$ is defined as $\Delta(x,y) := \sum_{(x)_i \neq (y)_i} 1$.

2.1 Parameterized Complexity

Since the development of the theory of NP-completeness, many computational problems of practical relevance were shown to be NP-hard. These problems are considered to be computational intractable, meaning they are widely believed not to be solvable by any algorithm running in time polynomial in the input size. Parameterized complexity theory aims at a more fine-grained analysis of such computationally hard problems. The goal is to identify certain *parameters* of the problem instances such that the presumably inherent super-polynomial part of the running time can be confined to the value of the parameter. This may allow to solve practical instances efficiently if the parameter is independent from the size of the instance and takes on a "small" value. Besides algorithmic benefits, the parameterized study of NP-hard problems provides a better understanding of the properties that make instances hard to solve.

Downey and Fellows [DF99] developed a parameterized complexity theory defining the fundamental notions and concepts. Their monograph focuses on structural complexity-theoretic results. Further complexity-theoretic approaches can be found in the book by Flum and Grohe [FG06]. Algorithmic approaches to parameterized problems are discussed in the book by Niedermeier [Nie06].

Parameterized Problems. Let Σ be a finite alphabet. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. The second component is called the *parameter* of the problem. A parameterized problem L is called *fixed-parameter tractable* with respect to the parameter if there exists an algorithm that, given an instance $(I, k) \in \Sigma^* \times \mathbb{N}$, decides whether $(I, k) \in L$ in time $f(k) \cdot |I|^{O(1)}$ for some computable function $f : \mathbb{N} \to \mathbb{N}$ only depending on k.

Search Tree Algorithms. One way to prove a parameterized problem fixed-parameter tractable is to give a *search tree algorithm* that solves it. The idea is to identify subsets of the input instance for which we know that they contain at least one element that contributes to an optimal solution. For each subset, we *branch* over all elements, that is, we simply try out all elements and recursively check whether we obtain a solution. If we can find such a subset in polynomial time and if the size of each subset as well as the recursion depth can be bounded by the parameter, then this procedure yields a fixed-parameter algorithm.

Problem Kernels. Another way to prove a parameterized problem L fixed-parameter tractable is to show that it admits a problem kernel. Indeed, it is known that a parameterized problem is fixed-parameter tractable if and only if it admits a problem kernel [CCDF97].

A reduction to a *problem kernel* is a mapping $r : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, $(I, k) \mapsto (I', k')$ such that $(I, k) \in L \Leftrightarrow (I', k') \in L$, and $|I'| \leq g(k)$, and $k' \leq h(k)$ holds for some computable functions $g, h : \mathbb{N} \to \mathbb{N}$ only depending on k. The function g is called the *size* of the problem kernel (I', k'). Moreover, the reduction r must be computable in

time polynomial in $|I| + k$. A problem kernel can be seen as a form of data reduction rule of guaranteed efficiency [LMS12, Bod09].

Parameterized Reductions. Let L and L' be two parameterized problems. A *parameterized (many-one) reduction* is a mapping $r : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N},\ (I, k) \mapsto (I', k')$ that is computable in $f(k) \cdot |I|^{O(1)}$ time for some computable function $f : \mathbb{N} \to \mathbb{N}$ solely depending on k such that $(I, k) \in L \Leftrightarrow (I', k') \in L'$ and $k' \leq g(k)$ for some computable function $g : \mathbb{N} \to \mathbb{N}$ only depending on k.

Parameterized Intractability. Since there are parameterized problems which are not known to be fixed-parameter tractable, there exist several parameterized complexity classes in order to characterize the different levels of intractability. Here, we introduce some of them which will appear throughout this work. For a comprehensive study, we refer to the books by Downey and Fellows [DF99] and Flum and Grohe [FG06]. To begin with, we introduce the classes

$$\text{FPT} \subseteq \text{W[1]} \subseteq \text{W[2]} \subseteq \text{W[3]} \subseteq \cdots \subseteq \text{XP}.$$

The class FPT contains all parameterized problems that are fixed-parameter tractable. For $t \in \mathbb{N}$, the class W[t] contains all parameterized problems that are parameterized many-one reducible to the satisfiability problem for boolean formulae of the form "products-of-sum-of-products ... of literals" with $t - 1$ alternations between products and sums, parameterized by the number of variables that are assigned true. The class XP contains all parameterized problems L for which it can be determined in $f(k) \cdot |I|^{g(k)}$ time whether $(I, k) \in \Sigma^* \times \mathbb{N}$ is in L for some computable functions f and g only depending on k. A parameterized problem L is considered intractable if it is W[1]-hard—that is if all problems in W[1] are parameterized many-one reducible to L. For example, Independent Set is W[1]-hard with respect to the parameter sought solution size.

> Independent Set
> **Input:** An undirected graph $G = (V, E)$ and a nonnegative integer k.
> **Question:** Is there a subset of vertices $I \subseteq V$ with k or more vertices that form and independent set, that is, I induces an edgeless subgraph of G?

Hitting Set parameterized by the sought solution size even is W[2]-hard. This means, that Hitting Set could be intractable even if Independent Set would be fixed-parameter tractable.

Hitting Set
Input: A finite universe U, a collection $\mathcal{C}$ of subsets of U, and a nonnegative integer k.
Question: Is there a subset $K \subseteq U$ with $|K| \leq k$ such that K contains at least one element from each subset in $\mathcal{C}$?

Bounded Fixed-Parameter Tractability. Instead of allowing arbitrary computable functions f in the definition of fixed-parameter tractability, one could also bound the growth of the parameter dependence. In doing so, one obtains the *bounded parameterized complexity theory* introduced by Flum et al. [FGW06, FG06]. For example, one could define f to be upper bounded by $2^{O(k)}$ (singly exponential). The subclass of FPT containing the parameterized problems which are solvable by a fixed-parameter algorithm running in such a time is called EPT.

EPT-Reductions. In order to define complexity classes for problems that are not bounded fixed-parameter tractable, one needs to define a proper notion of reduction: Let L and L' be two parameterized problems. An *ept-reduction* is a mapping $r : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N},\ (I, k) \mapsto (I', k')$ that is computable in $2^{O(k)} \cdot |I|^{O(1)}$ time such that $(I, k) \in L \Leftrightarrow (I', k') \in L'$ and $k' \in O(k + \log |I|)$.

Bounded Fixed-Parameter Intractability. As with the unbounded theory, there are also several classes containing bounded fixed-parameter intractable problems. Again, for each $t \in \mathbb{N}$, one can define a class EW[t]. The class EW[2] appearing in this work is defined to contain all parameterized problems that are ept-reducible to the satisfiability problem for boolean formulae of the form product-of-sum-of literals, parameterized by the number of variables that are assigned true.

The EW-classes allow for proofs of lower bounds in that for a parameterized problem to be EW[1]-hard basically means that it cannot be solved in time depending singly exponential on the parameter and polynomial on the input size. For example, Hitting Set is also known to be EW[2]-hard with respect to the sought solution size. The EW-classes are related to the W-classes in that, for $t \geq 2$, it holds that EW[t] = EPT implies W[t] = FPT.

2.2 Graphs

We shortly introduce some basic notions of graph theory. For further reading on graph theory, for example, see the book by Diestel [Die10].

Basic Definitions. A *simple undirected* graph $G = (V, E)$ consists of a set V of *vertices* and a set E of *edges*, where every edge $e \in E$ is a set such that $|e \cap V| = 2$. Usually, the number of vertices $|V|$ is denoted by n and the number of edges $|E|$ is denoted by m. A vertex $v \in V$ is *incident* with an edge $e = \{u, w\} \in E$ if $v \in e$. Two vertices are *adjacent* (or *neighbors*) if there is an edge $e \in E$ such that both are incident

with e. The incidence matrix $I_G \in \{0,1\}^{n \times m}$ of a graph G is a binary matrix defined to contain a 1 in the i-th row and j-th column if the vertex i is incident with edge j, otherwise it contains a 0. A graph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$ is called a *subgraph* of G. Let $V' \subseteq V$. The graph $G[V'] := (V', \{e \in E \mid e \subseteq V'\})$ is called the *(vertex-)induced* subgraph of G with respect to V'.

Some Special Graphs. A *path* of length k is a graph that is isomorphic to the graph

$$P_k = (\{v_1, \ldots, v_k\}, \{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\}\}).$$

A graph is called *complete* if it contains all possible edges. Such a graph is also called a *clique*. A *cluster graph* is a disjoint union of cliques. An equivalent characterization of a cluster graph is a graph that does not contain an induced P_3 (for example, see Shamir et al. [SST04]). An *independent set* is a set of vertices where no two vertices are adjacent. A *matching* is a graph where no two edges are incident with the same vertex. The *line graph* $L(G)$ of a graph G is defined to contain a vertex for every edge in G and two vertices are adjacent in $L(G)$ if and only if G contains a vertex that is incident with the corresponding edges.

3 Subspace Selection

This chapter deals with two problems belonging to the setting of subspace selection, namely Hidden Clusters and the Hidden Cluster Graph problem. As both of them will turn out to be NP-hard, we conduct a more detailed analysis in order to identify for which parameters the problems become fixed-parameter tractable and for which they remain hard to solve. section 3.1 studies Hidden Clusters followed by a discussion of the Hidden Cluster Graph problem in section 3.2.

3.1 Hidden Clusters

In this section, we focus on the Hidden Clusters problem by Charikar et al. [CGK+00]. Recall the problem definition:

Hidden Clusters
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ consisting of n points in d dimensions, $r \in \mathbb{Q}$, $\ell, k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that there exist ℓ centers $C = \{c_1, \ldots, c_\ell\} \subseteq \Sigma^d$ and an assignment of points to centers $\sigma : S \to C$ such that $\text{dist}_{|K}(x, \sigma(x)) \leq r$ for all points $x \in S$?

Charikar et al. [CGK+00] provided some (α, β)-approximation algorithms which approximate the radius r within a factor of α and the number of dimensions k within a factor of β^{-1}. More specifically, they gave a $(3, 1)$-approximation algorithm running in $O(n(\ell^3 + d)n^\ell d^{\binom{\ell}{2}})$ time for Hidden Clusters with the L_∞-norm as distance function and a randomized algorithm which yields an $(O(\log n), 1 + \epsilon)$-approximation with probability $1 - n^{-O(1)}$ for any $\epsilon > 0$ within $n^\ell 2^{\binom{\ell}{2}} d^{O(\log n)}$ time for the L_1-version. They proved that it is in fact NP-hard to find any $(\alpha, 1)$-approximation for the L_∞- and the L_1-version. Even with a constant number of centers ℓ, it is NP-hard, for any constants $\delta > 0$, $c > 1$, to obtain a $(2 - \delta, d^{1-\delta})$-approximation for the L_∞-version and a $(c, d^{1-\delta})$-approximation for the L_1-version.

The approximation hardness results for an arbitrary number of cluster centers are based on a special case of the Hidden Clusters problem, where Charikar et al. [CGK+00] considered a binary alphabet $\Sigma = \{0, 1\}$, a radius $r = 0$ and an arbitrary metric as distance function. We call this variant the Binary Hidden Clusters problem. The formal definition reads as follows:

Table 3.1: Overview of results for the Binary Hidden Clusters problem (new results are indicated by ▶).

Parameter†		Binary Hidden Clusters	Theorem
unparameterized	▷	NP-hard	[Theorem 3.2]
k	▶	W[1]-hard	[Corollary 3.3]
(ℓ, t)	▶	$O(\ell^t d(\ell^3 + n))$-time solvable	[Theorem 3.6]

† k: sought solution size, ℓ: number of cluster centers, t: number of dimensions to be deleted

Binary Hidden Clusters
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \{0,1\}^d$ consisting of n points in d dimensions and $\ell, k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that $|S_{|K}| \leq \ell$?

The results presented in this section all refer to this special case. Note that all hardness results automatically hold for the general formulation of the Hidden Clusters problem. We consider the following three parameters: The sought solution size k, the number of dimensions to be deleted $t := d - k$ (the *dual* parameter to k), and the number of cluster centers ℓ. Table 3.1 summarizes the results of this section.

3.1.1 NP- and W[1]-Hardness

Charikar et al. [CGK+00, Lemma 12] stated that it is NP-hard to find the optimal number of dimensions with any finite approximation on the radius (that is, an $(\alpha, 1)$-approximation) in polynomial time. In fact, their proof also implies general NP-hardness of Hidden Clusters. They described a reduction from Clique, however, without providing a detailed proof[1] of correctness. Based on their construction, we now give a full proof of NP-hardness. To this end, we first show the following technical lemma:

Lemma 3.1. *For $m, i \in \mathbb{N}$ with $m \geq 4$ and $1 \leq i \leq m$ it holds*

$$\binom{m-i}{2} < \binom{m}{2} - (i+1).$$

Proof. Let $m \geq 4$ and $1 \leq i \leq m$. We have to show the following:

$$i + 1 < \binom{m}{2} - \binom{m-i}{2}.$$

[1] After e-mail communication with Ravi Kumar in April 2012, it became apparent that there is no version containing a fully detailed proof.

The right hand side can be rewritten to

$$\binom{m}{2} - \binom{m-i}{2} = \sum_{j=1}^{i} \left[\binom{m-(j-1)}{2} - \binom{m-j}{2} \right] = \sum_{j=1}^{i} (m-j) = im - \binom{i+1}{2}.$$

Hence, for $i > 0$, we have to show the following inequality

$$\frac{i+1}{i} < m - \frac{i+1}{2}.$$

If $i = 1$, then the above inequality reads $3 < m$, which is true. The inequality also holds for $i = m$ since

$$\frac{m+1}{m} \leq \frac{5}{4} < \frac{3}{2} \leq \frac{m-1}{2}.$$

Finally, for $1 < i < m$ we have

$$\frac{i+1}{i} \leq \frac{3}{2} < 2 \leq m - \frac{i+1}{2},$$

which finishes the proof. □

Now, we show how Lemma 3.1 can be utilized to prove the main result.

Theorem 3.2. Binary Hidden Clusters *is* NP*-hard.*

Proof. We describe a polynomial-time many-one reduction from the Clique problem:

> Clique
> **Input:** An undirected graph $G = (V, E)$ and a nonnegative integer k.
> **Question:** Is there a subset $C \subseteq V$ of vertices of size at least k that forms a clique in G?

Let (G, k) be an instance of Clique with a simple graph $G = (V, E), |V| = n, |E| = m$ and $k \geq 0$. If $k \leq 3$ or $k \geq n - 2$, we simply solve the problem in polynomial time by trying out all $O(n^3)$ possible subsets of V of size k and return a trivial "yes"- or "no"-instance. To check if the chosen subset is a clique requires $O(n^2)$ time. Otherwise, consider the incidence matrix I_G of G with rows corresponding to vertices and columns corresponding to edges. Let (S, ℓ, k') be the Binary Hidden Clusters instance where $S \subseteq \{0,1\}^m$ is the set of all rows of I_G, the number of centers is $\ell = k + 1$ and $k' = \binom{k}{2}$. This instance can be computed in $O(nm)$ time. Thus, the overall time required to perform the reduction is in $O(n^5)$ and we claim that G has a clique of size k if and only if there is a subset $K \subseteq \{1, \ldots, m\}$ of *dimensions* with $|K| = \binom{k}{2}$ such that $|S_{|K}| \leq k + 1$.

To prove the correctness, we first assume that G contains a clique of size k. Then we can choose K to contain all dimensions corresponding to edges between vertices in the clique. Now, all vertices that are not in the clique correspond to the same point in $S_{|K}$ containing only 0's (the null point) because they are not incident with any edge corresponding to a dimension in K. So, all non-clique vertices correspond to the null

point. Together with the k points corresponding to the clique vertices this results in at most $k+1$ different points in $S_{|K}$.

Now, suppose that there is a subset K of dimensions with $|K| = \binom{k}{2}$ such that $|S_{|K}| \leq k+1$. Then we have to consider the cases where $S_{|K}$ contains the null point and where it does not. If $S_{|K}$ contains the null point, there are k points with at least one entry equal to 1. We call them *non-zero* points. Let N be the number of vertices in G that correspond to any of the k non-zero points in $S_{|K}$.

If $N = k$, then the $\binom{k}{2}$ edges corresponding to the chosen dimensions in K are induced by k vertices in G. This is tantamount with a clique of size k in G.

If $N > k$, then there exist two vertices u and v that correspond to the same point in $S_{|K}$. Note that this is only possible if both points contain a single 1 in the same dimension since G is a simple graph. This can only happen if the edge $\{u, v\}$ is *isolated in* K. Hereby, we refer to the situation that K contains the dimension corresponding to the edge $\{u, v\}$ and none of the dimensions corresponding to edges that are incident with u or v in G. Indeed, both endpoints of an isolated edge in K have the same point in $S_{|K}$ with a single entry equal to 1. It also follows that for each non-zero point in $S_{|K}$ there are at most two vertices in G corresponding to it. The number $i \leq k$ of edges isolated in K satisfies

$$\begin{aligned} & k - i = N - 2i \\ \Leftrightarrow \quad & i = N - k. \end{aligned}$$

Hence, there are $k - i$ vertices in G corresponding to $k - i$ non-zero points in $S_{|K}$ with $\binom{k}{2} - i$ edges between them. But $k - i$ vertices induce at most $\binom{k-i}{2}$ edges and Lemma 3.1 yields

$$\binom{k-i}{2} < \binom{k}{2} - i,$$

which shows that this is actually not possible.

Finally, we have to deal with the case that $S_{|K}$ contains only non-zero points and $|S_{|K}| \leq k+1$. But this situation is also not possible: Since all $n > k+2$ vertices in G correspond to a non-zero point in $S_{|K}$, there have to be $i = n - (k+1)$ edges isolated in K. Note that $2 \leq i \leq k+1$. Now we have $k+1-i$ vertices in G with $\binom{k}{2} - i$ edges between them. Again, by Lemma 3.1 it follows

$$\binom{k-(i-1)}{2} < \binom{k}{2} - i$$

and we end up with a contradiction. Hence, we have shown the correctness of the above reduction. □

Note that the above reduction runs in polynomial time and outputs a Binary Hidden Clusters instance with a sought solution size k' only depending on k. Thus, it is a parameterized reduction from Clique, which is known to be W[1]-complete with respect to the parameter k (see Downey and Fellows [DF99]). As a result, we obtain the following corollary:

Corollary 3.3. Binary Hidden Clusters *is* W[1]*-hard with respect to the parameter* k*.*

3.1.2 A Fixed-Parameter Algorithm

So far, we have seen that Binary Hidden Clusters is not only NP-hard but also W[1]-hard with respect to the parameter k. In contrast to these results, we now show that there is also a tractable case. As the goal of Hidden Clusters is to maximize the number of dimensions to keep, the parameter k typically will take on large values. Thus, it seems natural to seek for fixed-parameter algorithms with respect to the *dual* parameter $t := d - k$ (number of dimensions to be deleted) since it will take on small values accordingly. Another parameter of interest is the number of cluster centers ℓ as it is also conceivable that this will be small in some applications since one benefit of a cluster analysis is to get an overview of the main patterns or prototypes present in the data by grouping many individual samples into few clusters. In a simple scenario, the data can be grouped into two classes representing "positive" and "negative" examples of some phenomenon. For instance, one could imagine the data to represent some symptoms of patients in a medical study and the aim is to group the subjects into healthy and ill ones.

We show that the Binary Hidden Clusters problem is fixed-parameter tractable with respect to the combined parameter (ℓ, t). Since both parameters could be small in some cases, the combination (ℓ, t) constitutes a reasonable parameter. The concept of a *discriminating feature set* will be helpful in constructing a fixed-parameter algorithm for the Binary Hidden Clusters problem:

Definition 3.4. Let $S \subseteq \Sigma^d$. A *discriminating feature set* $D \subseteq \{1, \ldots, d\}$ of S is a subset of dimensions such that $\forall x, y \in S, x \neq y : x_{|D} \neq y_{|D}$.

Note that if we find a discriminating feature set for more than ℓ points in a given Binary Hidden Clusters instance, then we know that we have to delete at least one dimension out of it. We may not know which dimension to delete but if the discriminating feature set is not too large, then we could simply guess and try out all possibilities. Lemma 3.5 states an upper bound on the size of a discriminating feature set.

Lemma 3.5. *Let* $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ *be a set of* $n \geq 2$ *points. Then, there exists a discriminating feature set* $D \subseteq \{1, \ldots, d\}$ *of* S *of size at most* $n - 1$*.*

Proof. The proof is by induction on n: For $n = 2$ we only need one dimension to distinguish two different points. For the inductive step let $S = \{x_1, \ldots, x_{n+1}\} \subseteq \Sigma^d$ and let $S' \subset S$ be a subset of size n. From the induction hypothesis it follows that S' has a discriminating feature set D' of size $n - 1$. If D' is also a discriminating feature set of S, then we are done. Otherwise, since all points in $S'_{|D'}$ are distinct, there exists at most one point $y \in S'$ with $y_{|D'} = x_{|D'}$ for $x \in S \setminus S'$. But we know that there exists an $i \in \{1, \ldots, d\} \setminus D'$ with $(y)_i \neq (x)_i$ and thus $D := D' \cup \{i\}$ is a discriminating feature set of S of size n. □

Now we are ready to prove the following theorem:

Theorem 3.6. Binary Hidden Clusters *is solvable in* $O(\ell^t d(\ell^3 + n))$ *time.*

Proof. We describe a search tree algorithm that solves a given Binary Hidden Clusters instance (S, ℓ, k). At first, let $K = \{1, \ldots, d\}$. As long as $|S_{|K}| > \ell$, we choose an arbitrary subset $S' \subseteq S_{|K}$ with $|S'| = \ell + 1$ and find a discriminating feature set D of S' of size at most ℓ. Note that Lemma 3.5 ensures the existence of D. Moreover, the induction in the proof of Lemma 3.5 indicates a bottom-up procedure to determine D in $O(\ell^3 d)$ time. The set $S_{|K}$ can be computed in $O(nd)$ time by sorting the points lexicographically (for example with radix sort [Knu98]) and deleting multiple points afterwards. Now, it holds that K must not contain D because otherwise $|S_{|K}| > \ell$ still holds. So, we have to delete at least one dimension out of D from K. A simple branching over all dimensions in D yields an $O(\ell^t d(\ell^3 + n))$ search tree algorithm. □

So far, we have seen that Binary Hidden Clusters is fixed-parameter tractable for the combined parameter (ℓ, t). An interesting question—which we have to leave open here—is whether it is possible to prove Binary Hidden Clusters fixed-parameter tractable with respect to one of the two parameters alone. Furthermore, one could omit the restriction to a binary alphabet. The general case of Hidden Clusters with an arbitrary alphabet certainly constitutes a challenge for future research.

3.2 Hidden Cluster Graphs

We now proceed to clustering tasks where the actual number of centers is not known beforehand, which could often happen in practice, for example in an explorative setting, where there is no a priori knowledge about the underlying structure of the data at hand. In this case, we consider a different notion of clustering that compensates for the lack of information by defining the number of clusters implicitly. We already introduced the Hidden Cluster Graph problem:

Hidden Cluster Graph
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ consisting of n points in d dimensions, $r \in \mathbb{Q}$, $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that the graph $G_K = (V, E_K)$ with

$$V := S, \quad E_K := \{\{x_i, x_j\} \mid x_i \neq x_j \in V,\ \mathrm{dist}_{|K}(x_i, x_j) \leq r\}$$

is a cluster graph (that is, a union of disjoint cliques)?

Throughout this section, we consider the feature space $\Sigma^d = \mathbb{Q}^d$ equipped with the distance function $\mathrm{dist}^{(p)}(x, y) := \|x - y\|_p^p$ for $p \in \mathbb{N}$ or $\mathrm{dist}^{(\infty)}(x, y) := \|x - y\|_\infty$. We refer to this case as the L_p-Hidden Cluster Graph problem and give the following definition:

Table 3.2: Overview of results for the L_p-Hidden Cluster Graph problem (new results are indicated by ▶).

Parameter†		L_p-Hidden Cluster Graph	Theorem
unparameterized	▶	$O(d(n^2d + n^3))$-time solvable for $p = \infty$	[Proposition 3.7]
	▶	NP-hard for $1 \leq p < \infty$ and $\Sigma \supseteq \mathbb{N}$	[Corollary 3.9]
t	▶	W[2]-hard for $1 \leq p < \infty$ and $\Sigma \supseteq \mathbb{N}$	[Theorem 3.8]
(r, t)	▶	$O((2^p r)^t \cdot (n^2d + n^3))$-time solvable for $\Sigma \subseteq \mathbb{Z}$	[Theorem 3.10]

† t: number of dimensions to be deleted, r: radius

L_p-Hidden Cluster Graph
Input: A set $S = \{x_1, \ldots, x_n\} \subseteq \mathbb{Q}^d$ consisting of n points in d dimensions, $r \in \mathbb{Q}$, $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \geq k$ such that the graph $G_K = (V, E_K)$ with

$$V := S, \quad E_K := \{\{x_i, x_j\} \mid x_i \neq x_j \in V,\ \mathrm{dist}^{(p)}_{|K}(x_i, x_j) \leq r\}$$

is a cluster graph (that is, a union of disjoint cliques)?

It turns out that, in contrast to the Hidden Clusters problem—which is NP-hard for arbitrary metrics, the choice of distance function has a considerable impact on the tractability of the Hidden Cluster Graph problem. We show that L_p-Hidden Cluster Graph is NP-hard and W[2]-hard with respect to t for all finite $p \geq 1$ even if the dataset is restricted to contain natural numbers only, whereas for $p = \infty$, the problem is polynomial-time solvable. A summary of the results obtained in this section is given in Table 3.2.

3.2.1 A Polynomial-Time Algorithm

We begin our discussion of the Hidden Cluster Graph problem with a polynomial-time algorithm for the L_∞-distance function.

Proposition 3.7. *L_∞-Hidden Cluster Graph can be solved in $O(d(n^2d + n^3))$ time, assuming constant-time arithmetical operations.*

Proof. We describe a polynomial-time algorithm that deterministically solves a given L_∞-Hidden Cluster Graph instance (S, r, k). We start with the following observation: If we remove a dimension, the distance between two points induced by the L_∞-norm cannot increase. Since we are only allowed to select a subset of dimensions, this means that we can only decrease the distance between arbitrary points in S and not increase it. This corresponds to only adding edges to $G_{\{1,\ldots,d\}}$. In order to obtain a cluster

graph, we have to destroy all induced P_3's of the form $P = (\{u, v, w\}, \{\{u, v\}, \{v, w\}\})$ in $G_{\{1,\dots,d\}}$ by inserting the missing edge between u and w. Hence, we have to select a subset of dimensions K such that $\text{dist}_{|K}^{(\infty)}(u, w) = \max_{j \in K} |(u)_j - (w)_j| \leq r$. This can only be achieved by deleting all dimensions where u and w differ by more than r.

Now, our algorithm starts with the full set of dimensions $K = \{1, \dots, d\}$ and iteratively searches for an induced P_3 in G_K and deletes all dimensions of K that have to be deleted due to the above requirement. The algorithm terminates if there are no P_3's in G_K anymore or if $|K| < k$. In the former case it outputs "yes", whereas in the latter case it outputs "no".

algorithm 1 depicts the pseudocode. The computation of G_K can be done in $O(n^2 d)$ time. Hoàng et al. [HKSS12] showed that finding all induced P_3's in G_K can be done in $O(m^{1.5} + p_3(G_K))$ time, where m is the number of edges in G_K and $p_3(G_K)$ is the number of induced P_3's in G_K. The while-loop in line 3 is iterated at most d times since it always deletes at least one dimension in each step. Hence, the overall running time is in $O(d(n^2 d + n^3))$. □

Algorithm 1: L_∞-Hidden Cluster Graph

Input : Dataset $S \subseteq \mathbb{Q}^d$, radius $r \in \mathbb{Q}$, $k \in \mathbb{N}$
Output: Feature set $K \subseteq \{1, \dots, d\}$ such that G_K is a cluster graph

1 $K \leftarrow \{1, \dots, d\}$;
2 $G \leftarrow$ `compute` G_K;
3 **while** $\exists u, v, w \in V(G) : G[\{u, v, w\}] = (\{u, v, w\}, \{\{u, v\}, \{v, w\}\})$ **do**
4 $\quad K \leftarrow K \setminus \{j \in K \mid |(u)_j - (w)_j| > r\}$;
5 $\quad$ **if** $|K| < k$ **then return** `False`;
6 $\quad$;
7 $\quad G \leftarrow$ `compute` G_K;
8 **return** `True`(K);

3.2.2 NP- and W[2]-Hardness

We now show that the L_p-Hidden Cluster Graph problem is unlikely to be fixed-parameter tractable with respect to the number t of dimensions to be deleted if p takes on finite values. For this purpose, we describe a reduction from a problem called Lobbying occurring in the area of computational social choice. The original problem definition is due to Christian et al. [CFRS07]. They proved the W[2]-hardness of Lobbying parameterized by the number of modifications. However, we will use a slightly different problem formulation introduced by Bredereck et al. [BCH$^+$12], shown to be equivalent to the original one:

Lobbying
Input: A matrix $A \in \{0,1\}^{n \times m}$ and an integer $k > 0$.
Question: Can one modify (set to 1) at most k rows in A such that in the resulting matrix each column contains more 1's than 0's?

In their proof of W[2]-hardness Christian et al. [CFRS07] showed that the problem is in fact W[2]-hard with respect to the parameter k if the number of columns m is odd. But with m being odd, requiring more ones than zeros is equivalent to requiring at least as many ones as zeros. Moreover, exchanging ones with zeros and rows with columns clearly does not change the problem. Thus, we can formulate the following equivalent definition, which is more convenient for our purpose:

Lobbying*
Input: A matrix $A \in \{0,1\}^{m \times n}$ containing an odd number of columns n and an integer $k > 0$.
Question: Can one modify (set to 0) at most k columns in A such that in the resulting matrix each row contains at least as many 0's as 1's?

A reduction from Lobbying* might seem surprising at first glance since the problem originates from an area which is not directly related to cluster analysis. But a closer look on the above problem formulation reveals some similarity: The matrix A can be seen as a set of points in an n-dimensional feature space. If the columns of the matrix A are interpreted as dimensions, then modifying columns basically means deleting dimensions of a dataset. Moreover, the goal is to reduce the number of 1's in each row, which corresponds to reducing the distance between points. The following proof demonstrates in detail how Lobbying* can be turned into a L_p-Hidden Cluster Graph problem by choosing an appropriate dataset and radius.

Theorem 3.8. *L_p-Hidden Cluster Graph is W[2]-hard with respect to the parameter t for all $1 \leq p < \infty$.*

Proof. We give a parameterized many-one reduction from Lobbying*: Let (A, k) be an instance of Lobbying* with $A \in \{0,1\}^{m \times n}$ containing m rows $a_1, \ldots, a_m \in \{0,1\}^n$. We assume that every row of A contains more ones than zeros because otherwise we could delete it from the input without changing the answer to the question. We define an L_p-Hidden Cluster Graph instance (S, r, k') with

$$S := \bigcup_{1 \leq i \leq m} \{u_i, v_i, w_i\} \subseteq \mathbb{Q}^n, \quad r := 2^{p-1}n, \quad k' := n - k.$$

The idea of the construction is to let S contain three data points u_i, v_i and w_i for every row a_i in A such that their induced subgraph $H_i := G_{\{1,\ldots,n\}}[\{u_i, v_i, w_i\}]$ is a P_3, that is

$$H_i = (\{u_i, v_i, w_i\}, \{\{u_i, v_i\}, \{v_i, w_i\}\}).$$

This can be achieved by setting

$$u_1 := \vec{0}, \qquad w_1 := 2a_1, \qquad v_1 := \frac{u_1 + w_1}{2},$$
$$u_i := w_{i-1} + 2\vec{n}, \qquad w_i := u_i + 2a_i, \qquad v_i := \frac{u_i + w_i}{2}, \qquad i = 2, \ldots, m,$$

where $\vec{x} := (x, \ldots, x) \in \Sigma^n$ for $x \in \Sigma$. The above construction only requires the feature space $\mathbb{N}^n$ in order to be well-defined. The reduction can be done using $O(mn)$ arithmetical operations. It is a parameterized reduction since $t = n - k' = k$. Figure 3.1 illustrates the constructed dataset. Now, for all $i = 1, \ldots, m$ we have

$$\mathrm{dist}^{(p)}(u_i, w_i) = \|2a_i\|_p^p = 2^p \sum_{j=1}^{n} |(a_i)_j|^p \geq 2^p \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right) > r$$

and

$$\mathrm{dist}^{(p)}(u_i, v_i) = \mathrm{dist}^{(p)}(v_i, w_i) = \|a_i\|_p^p \leq n \leq r.$$

Since $G_{\{1,\ldots,n\}}$ is defined to contain an edge between two vertices if and only if the distance of their corresponding points in S is at most r, it follows indeed that H_i is a P_3. By construction, the H_i are independent of each other in the sense that, for every non-empty subset $K \subseteq \{1, \ldots, n\}$ of dimensions, G_K never contains an edge between any vertices from H_i and H_j for $i \neq j$. To verify this, we first claim that, by construction, the smallest distance between any vertices from H_i and H_j is the distance of w_i and u_j: Let $1 \leq i < j \leq m$, the following identities hold

$$\begin{aligned}
v_j - w_i &= u_j - w_i + a_j, & u_j - v_i &= u_j - w_i + a_i,\\
w_j - w_i &= u_j - w_i + 2a_j, & u_j - u_i &= u_j - w_i + 2a_i,\\
v_j - v_i &= u_j - w_i + a_j + a_i, & w_j - v_i &= u_j - w_i + 2a_j + a_i,\\
v_j - u_i &= u_j - w_i + 2a_j + a_i, & w_j - u_i &= u_j - w_i + 2a_j + 2a_i,
\end{aligned}$$

and

$$u_j - w_i = (j - i) \cdot \vec{n} + 2 \sum_{k=1}^{j-i-1} a_{i+k}.$$

Note that all components of $u_j - w_i$, a_i and a_j are positive or at least zero. The L_p-norms of all the above differences are thus greater or equal to $\|u_j - w_i\|_p$. Accordingly, all distances between vertices from H_i and H_j are greater or equal to $\mathrm{dist}^{(p)}_{|K}(u_j, w_i)$. For every non-empty subset of dimensions K this distance is bounded from below by

$$\begin{aligned}
\mathrm{dist}^{(p)}_{|K}(u_j, w_i) &= \mathrm{dist}^{(p)}_{|K}\Big(w_i + (j - i) \cdot 2\vec{n} + 2 \sum_{k=1}^{j-i-1} a_{i+k}, w_i\Big)\\
&= \sum_{l \in K} \Big| \Big(w_i + (j - i) \cdot 2\vec{n} + 2 \sum_{k=1}^{j-i-1} a_{i+k}\Big)_l - (w_i)_l \Big|^p\\
&\geq 2^p \sum_{l \in K} |(\vec{n})_l|^p = 2^p \cdot |K| \cdot n \geq 2^p n > r.
\end{aligned}$$

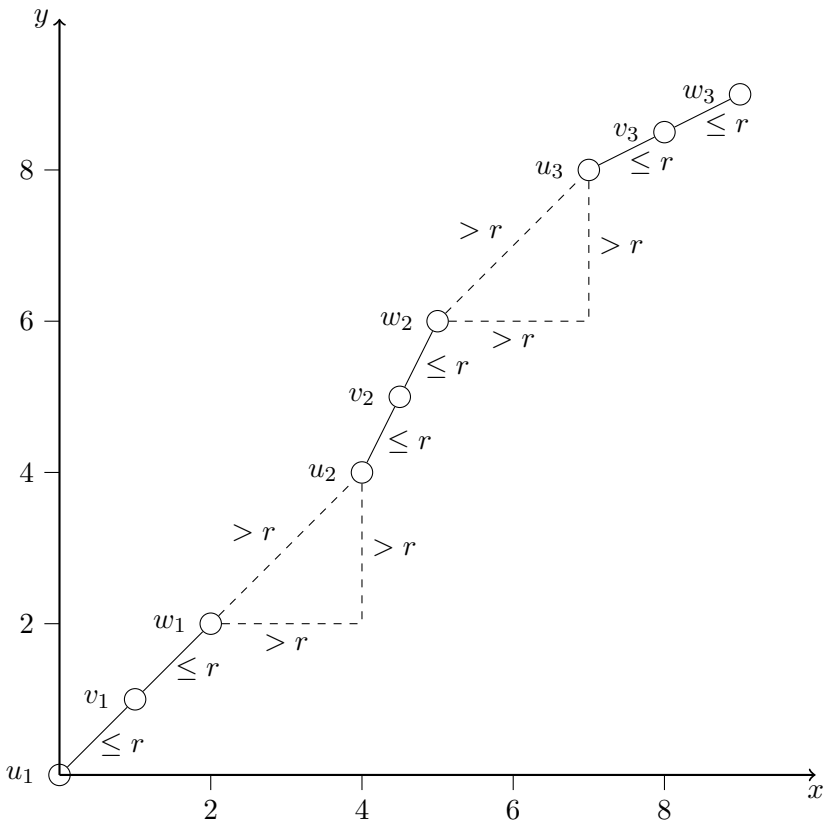

Figure 3.1: A two-dimensional illustration of the constructed L_p-Hidden Cluster Graph instance: For each row a_i in the lobbying matrix A there are three points u_i, v_i, w_i in the dataset S such that, for every non-empty subset of dimensions K, they induce a P_3 in G_K. This is achieved by recursively setting $v_i = u_i + a_i$, $w_i = v_i + a_i$ and choosing an appropriate radius $\|a_i\|_p^p \leq r < \|2a_i\|_p^p$. The picture indicates three such P_3's. For example, the distance between u_1 and v_1 is the same as the distance between v_1 and w_1 and both are smaller than the radius r, whereas u_1 and w_1 have a distance greater than r. Accordingly, there are edges from u_1 to v_1 and from v_1 to w_1 but not from u_1 to w_1. Note that the nearest point to any of the points in the first P_3 is u_2. Its distance to w_1 is greater than r in every dimension, which ensures that there is no edge between w_1 and u_2. The construction is such that this holds for all w_i and u_{i+1}.

Thus, there will never be an edge in G_K between vertices from H_i and H_j for any K. The only solution of this instance is the cluster graph consisting of the m disjoint triangles obtained by inserting the missing edge in each H_i. In order to insert the missing edge between u_i and w_i in every H_i, we have to find a subset of dimensions K such that

$$\operatorname{dist}^{(p)}_{|K}(u_i, w_i) = 2^p \sum_{j \in K} |(a_i)_j|^p \leq r = 2^{p-1} n$$

holds for all $i = 1, \ldots, m$. In other words, we have to delete at most t dimensions (that is, set entries in a_i to zero) such that for the remaining dimensions K it holds

$$\sum_{j \in K} |(a_i)_j|^p \leq \frac{n}{2}.$$

Since a_i is a binary point, the above equation states that the modified a_i contains at least as many zeros as ones, which is exactly the Lobbying* problem. So, the L_p-Hidden Cluster Graph instance is a "yes"-instance if and only if the initial Lobbying* instance is a "yes"-instance. □

As a byproduct, the above reduction also yields NP-hardness since it runs in polynomial time and Lobbying is NP-hard (for example, see Bredereck et al. [BCH+12]).

Corollary 3.9. *L_p-Hidden Cluster Graph is NP-hard for every $1 \leq p < \infty$.*

It is worth mentioning that the proof of Theorem 3.8 relies on some assumptions that could be considered unrealistic or pathological. For example, the data points of the constructed instance are very scattered, which yields a clustering with many clusters each of which contains only a few points. As a consequence, the reduction requires the data points and the radius to take on unbounded values. The next subsection addresses this issue and presents an approach towards fixed-parameter tractability.

3.2.3 A Fixed-Parameter Algorithm

We already remarked that the reduction from the proof of Theorem 3.8 requires the data points to take on unbounded values. Moreover, the *diameter* δ, that is the maximum distance between any two points in S, of the dataset can become arbitrary large. This subsection shows that an unbounded diameter is indeed necessary for L_p-Hidden Cluster Graph to be W[2]-hard with respect to the parameter t. To this end, observe that the radius r can always be bounded from above by the diameter δ since otherwise the graph $G_{\{1,\ldots,d\}}$ is a clique and thus a cluster graph, which corresponds to a trivial "yes"-instance. It is therefore sufficient to prove the following theorem:

Theorem 3.10. *If all points are restricted to integer coordinates, L_p-Hidden Cluster Graph is $O((2^p r)^t \cdot (n^2 d + n^3))$-time solvable for $p \geq 1$ under the assumption of constant-time arithmetic operations.*

Proof. We solve a given instance (S, r, k) of L_p-Hidden Cluster Graph by the following algorithm: Start with the full set of dimensions $K = \{1, \ldots, d\}$. If G_K is not a cluster graph, it contains an induced P_3, say $P = (\{u, v, w\}, \{\{u, v\}, \{v, w\}\})$. Since we have to select a subset of dimensions, distances between points can only be decreased. Thus, in order to get a cluster graph, we have to select a subset $K' \subset K$ such that $\{u, v, w\}$ does not induce a P_3 in $G_{K'}$, that is, $G_{K'}$ contains the edge $\{u, w\}$. Recall that, by definition of G_K, we have

$$\mathrm{dist}^{(p)}_{|K}(u, v) \leq r, \quad \mathrm{dist}^{(p)}_{|K}(v, w) \leq r, \quad \mathrm{dist}^{(p)}_{|K}(u, w) > r.$$

Moreover, as $\|\cdot\|_p$ is a norm, the triangle inequality yields

$$\mathrm{dist}^{(p)}_{|K}(u, w) = \|u - w\|_p^p \leq (\|u - v\|_p + \|v - w\|_p)^p \leq 2^p r.$$

By substitution of the definition of $\mathrm{dist}^{(p)}_{|K}$, we get

$$\mathrm{dist}^{(p)}_{|K}(u, w) = \sum_{j \in K} |(u)_j - (w)_j|^p \leq 2^p r.$$

Since all points consist of integer coordinates only, each summand of the above sum is either zero or at least one. It follows that K contains at most $2^p r$ dimensions in which u and w differ and we have to delete at least one of them. A simple branching over all feasible dimensions yields a search tree of size $O((2^p r)^t)$. Computing G_K and finding P_3's requires $O(n^2 d + n^3)$ time. The overall running time is $O((2^p r)^t \cdot (n^2 d + n^3))$. □

As we have seen, L_p-Hidden Cluster Graph parameterized by the radius and the number of dimensions to be deleted is fixed-parameter tractable. There might exist even more parameterizations allowing for fixed-parameter tractability: For example, one could be interested in the case that the data points contain only a finite number of different values. With an upper bound on the coordinates, the proof of Theorem 3.8 does not work. This may indicate that the problem is tractable with respect to this parameter. Another parameter, for which we could not provide any results so far, is the sought solution size k. Since the problem is already known to be W[2]-hard with respect to its dual parameter t, it would be interesting to know whether the problem is also hard with respect to k or not. Moreover, the algorithm from Theorem 3.10 requires points with integer coordinates. It is thus obvious to ask whether the problem remains fixed-parameter tractable if we omit this restriction and allow arbitrary rational numbers. These are some possible questions to be addressed in future work.

4 Dimension Reduction

In this chapter we study the Distinct Vectors problem where the goal is to select a minimum number of dimensions such that all given points can still be distinguished from each other within the selected dimensions.

Distinct Vectors
Input: A multiset $S = \{x_1, \ldots, x_n\} \subseteq \Sigma^d$ of n distinct points in d dimensions and $k \in \mathbb{N}$.
Question: Is there a subset $K \subseteq \{1, \ldots, d\}$ of dimensions with $|K| \leq k$ such that all points in $S_{|K}$ are still distinct?

Keep in mind that throughout this chapter S and $S_{|K}$ are defined to be multisets allowing for multiple elements. Charikar et al. [CGK+00] showed that, unless P = NP, there is a constant $c > 0$ such that the Distinct Vectors problem is hard to approximate within a factor of $c \cdot \log d$.

We analyze various cases of the Distinct Vectors problem and contribute further hardness results as well as algorithms for some tractable cases. Our considerations are based on the following parameters: The size of the alphabet σ, the sought solution size k, the number of dimensions to be deleted $t := d - k$ and the maximum Hamming distance between any pair of points f.

In section 4.1 we focus on the case where the dataset consists of points over a binary alphabet ($\sigma = 2$). We prove NP-hardness and several parameterized hardness results such as EW[2]-hardness with respect to the combination of requested solution size and alphabet size and W[1]-hardness with respect to the number of dimensions to be deleted. Further, we consider an even more specific case where the points are restricted not to be "*too different*" from each other in the sense that the number of dimensions in which two points differ (the Hamming distance) is bounded from above by a constant f. For this case the following dichotomy holds: Distinct Vectors with a binary alphabet is polynomial-time solvable for $f \leq 3$ and NP-hard for $f > 3$.

The general case of an alphabet with arbitrary size is treated in section 4.2. We show that—besides the hardness results established in section 4.1—there are parameterizations for which Distinct Vectors is fixed-parameter tractable. For example, we provide problem kernels with respect to the sought solution size in combination with the size of the alphabet and in combination with the maximum pairwise Hamming distance. Intuitively, an alphabet of unbounded size, however, does not make the problem easier. Accordingly, we prove that, for alphabets of arbitrary cardinality, Distinct Vectors is W[2]-hard with respect to the sought solution size k. A summary of the results obtained in this chapter is given in Table 4.1.

Table 4.1: Overview of results for the Distinct Vectors problem (new results are indicated by ▶).

Parameter†		Distinct Vectors	Theorem
unparameterized	▷	NP-hard for $\sigma = 2$	[Theorem 4.1]
	▶	and constant $f \geq 4$	[Theorem 4.4]
	▶	$O(n^3 d)$-time solvable for $\sigma = 2$ and $f = 3$	[Theorem 4.3]
	▶	$O(n^2 d)$-time factor-f approximable	[Proposition 4.9]
k	▶	W[2]-hard	[Theorem 4.10]
t	▶	W[1]-hard for $\sigma = 2$ and $f \geq 4$	[Corollary 4.5]
(k, f)	▶	$O(f^k \cdot nd)$-time solvable	[Proposition 4.8]
	▶	$O((f! \cdot f^{f+1} \cdot (k+1)^f)^2)$-size kernel in $O(n^2(d + f \log f + \log n) + f(n^4 + d))$ time	[Theorem 4.7]
(k, σ)	▶	EW[2]-hard	[Theorem 4.1]
	▶	$O(\sigma^{\sigma^k + k})$-size kernel in $O(d^2 n)$ time	[Proposition 4.6]

† σ: alphabet size, k: sought solution size, t: number of dimensions to be deleted, f: maximum number of dimensions in which any pair of points differs

4.1 Distinct Vectors on a Binary Alphabet

In this section, we concentrate on Distinct Vectors instances with binary data. Based on a proof by Charikar et al. [CGK+00], we show that even the binary case of Distinct Vectors is NP-hard. The same proof yields EW[2]-hardness for the parameters sought solution size k and alphabet size σ. This result implies a lower bound on the running time for any fixed-parameter algorithm. As we will see in the next section, there is a problem kernel with respect to the size of the alphabet and the size of the sought solution, which proves the problem to be fixed-parameter tractable.

The second part of this section deals with an even more restricted version of Distinct Vectors with a bounded number of dimensions in which any two points differ (the Hamming distance). We show that Distinct Vectors with binary data is NP-hard even if this bound is four. For smaller bounds, however, the problem turns out to be polynomial-time solvable.

4.1.1 NP- and EW[2]-Hardness

Charikar et al. [CGK+00, Theorem 20] proved that Distinct Vectors is NP-hard to approximate in polynomial time within a factor of $c \log d$. Their proof also implies NP-hardness, for they used a polynomial-time many-one reduction from the NP-hard

Set Cover problem. We provide a different proof of this result using an adapted reduction from Hitting Set, where we basically exchange rows and columns of the element-set incidence matrix of the Set Cover instance. The effort pays off because the reduction from Hitting Set allows us to conclude even more hardness results such as EW[2]-hardness, for example.

Theorem 4.1. Distinct Vectors *is* NP*-hard even for a binary alphabet and* EW[2]*-hard with respect to the combined parameter sought solution size k and alphabet size σ.*

Proof. We give a reduction from Hitting Set:

Hitting Set
Input: A finite universe U, a collection $\mathcal{C}$ of subsets of U, and a nonnegative integer k.
Question: Is there a subset $K \subseteq U$ with $|K| \leq k$ such that K contains at least one element from each subset in $\mathcal{C}$?

Hitting Set is known to be NP-hard and EW[2]-hard with respect to k (see Flum and Grohe [FG06]). Given an instance $(U, \mathcal{C}, k)$ of Hitting Set with $U = \{u_1, \ldots, u_m\}$, $\mathcal{C} = \{C_1, \ldots, C_n\}$, we construct a Distinct Vectors instance (S, k') with

$$S := \{x_1, \ldots, x_n, x'_1, \ldots, x'_n\} \subseteq \{0,1\}^{m+\lceil \log n \rceil},$$
$$k' := k + \lceil \log n \rceil,$$

where

$$(x_i)_j := \begin{cases} 1, & u_j \in C_i \\ 0, & u_j \notin C_i \end{cases} \quad \text{and} \quad (x'_i)_j := 0$$

for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. In the last $\lceil \log n \rceil$ dimensions we define x_i and x'_i to contain the binary representation of $i - 1$. This reduction is computable in $O(n(m + \log n))$ time. Moreover, note that the parameters $k' = k + \lceil \log n \rceil$ and $\sigma = 2$ meet the conditions of an ept-reduction. Thus, we have a polynomial-time many-one reduction and an ept-reduction at once. It remains to show the correctness.

Let $K \subseteq U$ be a solution of $(U, \mathcal{C}, k)$ with $|K| \leq k$. Then we can solve (S, k') by choosing the subset of dimensions K' that contains all dimensions corresponding to elements in K and the last $\lceil \log n \rceil$ dimensions. This choice allows us to distinguish each x_i from x'_i since K contains at least one element from each C_i, which ensures that there is at least one 1-entry in every $x_{i|K'}$. Furthermore, picking the last $\lceil \log n \rceil$ dimensions enables us to distinguish each $x_{i|K'}$ from $x_{j|K'}$ and each $x'_{i|K'}$ from $x'_{j|K'}$ for $i \neq j$. Since $|K'| \leq k'$, it follows that K' is a solution for (S, k').

Conversely, assume that K' with $|K'| \leq k'$ is a solution for (S, k'). Then K' contains the last $\lceil \log n \rceil$ dimensions because otherwise it is not possible to distinguish $x'_{i|K'}$ from $x'_{j|K'}$ for $i \neq j$. Note that x_i and x'_i cannot be distinguished by the last $\lceil \log n \rceil$ dimensions. Thus, K' contains at most k of the first m dimensions which ensure that each x_i can be distinguished from x'_i, that is every $x_{i|K'}$ contains at least one 1. This implies a solution of the original instance $(U, \mathcal{C}, k)$. Hence, the above reduction is correct. □

Notice that the reduction given above is not a parameterized many-one reduction since the sought solution size depends not only on k but also on n (more precisely $\log n$). Thus, we cannot infer from the W[2]-hardness of Hitting Set (see the book by Downey and Fellows [DF99]) that Distinct Vectors is W[2]-hard with respect to k for a binary alphabet. In section 4.2, we show that Distinct Vectors is in fact fixed-parameter tractable with respect to k for any alphabet of constant size.

One may argue that the above reduction produces a rather artificial Distinct Vectors instance because it contains "dummy" points x'_i which mainly consist of 0's and it numbers all points in the last $\lceil \log n \rceil$ dimensions. Nevertheless, we show that Distinct Vectors is also hard to solve on instances appearing more "natural". The following subsection contains another proof of NP-hardness, for an even more restricted variant, where the resulting instance basically takes on the form of an incidence matrix of arbitrary graphs.

4.1.2 Bounded Pairwise Hamming Distance: A Dichotomy

In this subsection, we further restrict our considerations to instances with points of bounded "*degree of distinctiveness*". Hereby, we refer to instances where each pair of points differs in at most f dimensions. In other words, the Hamming distance of any pair of points is bounded from above by f. For example, this situation can arise for *sparse* datasets where the points mainly contain 0's. Intuitively, if the dataset consists of points that are all "similar" to each other, one could hope to be able to solve the instance efficiently since there are at most f dimensions to choose from in order to distinguish two points. Later on, however, we will see that this is only the case for very small values of f: Even for $f = 4$, Distinct Vectors is NP-hard.

But first, we address the value of f that makes the problem tractable: We show that Distinct Vectors is in fact solvable in polynomial time if the pairwise Hamming distance is upper bounded by three. To start with, we introduce the following combinatorial lemma, which we use to identify the polynomial-time solvable special cases of Distinct Vectors.

Lemma 4.2. *Let $m, n \in \mathbb{N}$ with $m > n + 1$ and let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be a family of pairwise different sets, each of size n, such that*

$$\forall A_i \neq A_j : |A_i \cap A_j| = n - 1.$$

Then it follows

$$\forall A_i \neq A_j : A_i \cap A_j = \bigcap_{k=1}^{m} A_k.$$

The structure formed by the A_i is known as a *sunflower* [Juk11].

Proof. We assume $n \geq 2$ since the statement trivially holds for $n = 1$. We define $B_{ij} := A_i \cap A_j$ with $|B_{ij}| = n - 1$ for all $i \neq j$. Notice that it is sufficient to show $B_{12} = B_{13} = \ldots = B_{1m}$ because this already implies the lemma. We will prove this by contradiction.

Suppose that there exist indices i, j with $2 \leq i < j \leq m$ such that $B_{1i} \neq B_{1j}$. Then $|B_{1i} \cap B_{1j}| = n - 2$ and thus $B_{ij} \setminus A_1 =: C$ is non-empty and consists of one element. We claim that, for all $k = 2, \ldots, m$, it holds that $C \subset A_k$. This can be seen as follows: By assumption, the sets B_{ik} and B_{jk} contain $n - 1$ elements. If A_k does not contain C, then it follows

$$B_{ik} = A_i \setminus C = B_{1i} \quad \text{and} \quad B_{jk} = A_j \setminus C = B_{1j}.$$

We obtain

$$A_k = B_{ik} \cup B_{jk} = B_{1i} \cup B_{1j} = A_1,$$

which is a contradiction. Thus, A_k can be written as $A_k = C \cup B_{1k}$. Since $A_2, \ldots, A_m$ are all pairwise different, it follows that $B_{12}, \ldots, B_{1m}$ are all pairwise different. But now we have $m - 1 > n$ different subsets of A_1 of size $n - 1$, which is not possible since A_1 is of size n. □

Theorem 4.3. Distinct Vectors *is solvable in* $O(n^3 d)$ *time for a binary alphabet and* $f \leq 3$.

Proof. We give a search tree algorithm that solves a given Distinct Vectors instance (S, k). The restriction $f = 3$ guarantees that there are not "too many" branches to consider. For $x \in S$ and $i \in \mathbb{N}$ we define the following sets:

$$D_x := \{j \in \{1, \ldots, d\} \mid (x)_j = 1\},$$
$$S_i := \{x \in S \mid |D_x| = i\}.$$

Herein, D_x denotes the subset of dimensions in which x equals 1. By S_i we refer to the subset of points which are equal to 1 in exactly i dimensions. Without loss of generality, we can assume that $\vec{0} \in S$. If this is not the case, then we can simply fix an arbitrary point $x_0 \in S$ and swap 1's and 0's in all points in S in all dimensions where x_0 equals 1. This yields in linear time an equivalent instance with $x_0 = \vec{0} \in S$.

Let (S, k) be an instance of Distinct Vectors with $|S| = n$ binary points in d dimensions. The bound $f = 3$ implies that each point in S contains at most three 1's since otherwise it differs in more than three dimensions from $\vec{0}$. Thus, we can partition the dataset S as follows:

$$S = \{\vec{0}\} \uplus S_1 \uplus S_2 \uplus S_3.$$

Moreover, the restriction $f = 3$ also implies the following two conditions, which constitute the crucial points for our proof.

$$\forall x, y \in S_3 : |D_x \cap D_y| = 2 \tag{4.1}$$
$$\forall x, y \in S_2 : |D_x \cap D_y| = 1 \tag{4.2}$$

The algorithm starts with considering the subset S_3. The points in S_3 can only be distinguished from each other by a subset of the dimensions

$$D^3 := \bigcup_{x \in S_3} D_x.$$

x_1	1	1	1				
x_2	1	1		1			
x_3	1	1			1		
x_4	1	1				1	
x_5	1	1					1

Figure 4.1: The set S_3 represented as a matrix with rows corresponding to points. The columns correspond to the dimensions in D^3. Zero entries are omitted. Each pair of points has to share a 1 in two dimensions. With more than four points this is only possible if there are two dimensions in which all points are 1. Any solution contains at least all but one of the other dimensions.

If $|S_3| \leq 4$, we simply branch over all possible subsets of D^3. With a constant number of at most four distinct points in S_3, the size of D^3 is also bounded by a constant and so there are only constantly many subsets to try out. If $|S_3| > 4$, then statement (4.1) together with Lemma 4.2 implies that

$$C^3 := \bigcap_{x \in S_3} D_x$$

contains two dimensions. It follows that for each dimension $j \in D^3 \setminus C^3$ there exists exactly one point $x \in S_3$ with $(x)_j = 1$. This situation is depicted in Figure 4.1. Any solution has to contain all but one dimension from $D^3 \setminus C^3$ because otherwise there are two points that cannot be distinguished. Hence, we can try out all subsets of $D^3 \setminus C^3$ of size at least $|S_3| - 1$. Together with the four possible subsets of C^3 we end up with at most $4(n+1)$ subsets of D^3 to branch over.

Now, we continue with the subset S_2. The points in this subset can only be distinguished from one another by some of the dimensions contained in the following set:

$$D^2 := \bigcup_{x \in S_2} D_x.$$

If $|S_2| \leq 3$, we simply branch over all $O(1)$ subsets of D^2. For $|S_2| > 3$, it follows by statement (4.2) and Lemma 4.2 that

$$C^2 := \bigcap_{x \in S_2} D_x$$

contains one dimension. Figure 4.2 shows such a dataset. In order to distinguish all points in S_2 from each other, any solution has to contain at least $|S_2| - 1$ dimensions from $D^2 \setminus C^2$. This results in at most $2(n+1)$ subsets of D^2 to consider.

Finally, for S_1 it is necessary to select all dimensions in

$$D^1 := \bigcup_{x \in S_1} D_x.$$

x_1	1	1			
x_2	1		1		
x_3	1			1	
x_4	1				1

Figure 4.2: The points in S_2 represented as rows of a binary matrix with columns corresponding to the dimensions in D^2. Zero entries are omitted. Each pair of points has to share a 1 in one dimension. For more than three points there exists a dimension in which all points are 1. Any solution contains at least all but one of the other dimensions.

in order to distinguish the points from $\vec{0}$.

Thus, we end up with at most $4(n+1) \cdot 2(n+1) \in O(n^2)$ possible subset selections. For each selection we have to check whether it is a solution or not. This can be done in $O(nd)$ time by sorting the dataset lexicographically with radix sort [Knu98] and comparing successive points. This gives a search tree algorithm with an overall running time of $O(n^3 d)$. □

We now move on to the case $f > 3$. Now the condition 4.2 from the proof above does not hold and therefore we cannot apply Lemma 4.2, which is crucial in that it guarantees a regular structure of the dataset that makes the instance easy to solve. Instead, we will shortly see that the dataset can now "encode" arbitrary graphs.

For concreteness, we claim that if a pair of points is allowed to take on different values in at least four dimensions, then the problem becomes indeed NP-hard. To prove this, we describe a polynomial-time many-one reduction from a special variant of the well-known Independent Set problem. We refer to this variant as Distance-3 Independent Set. It is defined as follows:

> Distance-3 Independent Set
> **Input:** An undirected graph $G = (V, E)$ and a nonnegative integer k.
> **Question:** Is there a subset of vertices $I \subseteq V$ of size at least k such that any pair of vertices from I has distance at least three?

Here, the distance of two vertices is the number of edges contained in the shortest path between them. Note that the classic Independent Set problem is the same as Distance-2 Independent Set. Distance-3 Independent Set can be shown to be NP-hard by a reduction from the Induced Matching problem.

> Induced Matching
> **Input:** An undirected graph $G = (V, E)$ and a nonnegative integer k.
> **Question:** Does G contain a matching of size at least k that forms an induced subgraph of G?

Cameron [Cam89] proved this problem to be NP-hard. The reduction to Distance-3 Independent Set is simple and based on the observation that a graph G contains

an induced matching of size k if and only if there exists a distance-3 independent set of size k in the line graph $L(G)$ (for example, see the work by Brandstädt and Mosca [BM11]). After this preliminary considerations, we show how to reduce Distance-3 Independent Set to Distinct Vectors.

Theorem 4.4. Distinct Vectors *is* NP*-hard for a binary alphabet and* $f \geq 4$.

Proof. Let $(G = (V, E), k)$ with $|V| = n$ and $|E| = m$ be an instance of Distance-3 Independent Set and let Z be the $m \times n$ transposed incidence matrix of G with rows corresponding to edges and columns to vertices. The dataset S of our Distinct Vectors instance (S, k') is defined to contain all m rows of Z and the null vector. The sought solution size is set to $k' := n - k$. Notice that each point in S contains exactly two 1's (except the null vector). Thus, each pair of points differs in at most $f = 4$ dimensions. The instance (S, k') can be computed in $O(nm)$ time.

Correctness follows by the following argument: The subset $I \subseteq V$ is a solution of (G, k) if and only if it is of size k and every edge in G has at least one endpoint in $V \setminus I$ and no vertex in $V \setminus I$ has two neighbors in I. In other words, the latter condition says that no two edges with an endpoint in I share the same endpoint in $V \setminus I$. Equivalently, for the subset K of dimensions corresponding to the vertices in $V \setminus I$, it holds that all rows of Z in $S_{|K}$ contain at least one 1 and no two points contain only a single 1 in the same dimension. This holds if and only if K is a solution for (S, k') because S contains the null vector and thus two points can only be identical in $S_{|K}$ if either they consist of 0's only or contain a single 1 in the same dimension. □

As a last point, we remark that Induced Matching is not only NP-hard but also W[1]-hard with respect to the parameter k as was shown by Moser and Thilikos [MT09]. From this result we can infer that Distance-3 Independent Set is also W[1]-hard with respect to k. To see why this is of interest for us, notice that the reduction given in the proof of Theorem 4.4 yields an instance where the number of dimensions to be deleted is $t := n - k' = k$. Thus, we have a parameterized reduction from Distance-3 Independent Set parameterized by k to Distinct Vectors parameterized by t and the following corollary holds:

Corollary 4.5. Distinct Vectors *is* W[1]*-hard with respect to the number of dimensions to delete.*

4.2 Distinct Vectors on an Arbitrary Alphabet

As we have seen in the last section, Distinct Vectors is NP-hard and W[1]-hard with respect to the number of dimensions to be deleted even in the case of a binary alphabet where the pairwise Hamming distance of the points is bounded by four. Although, of course, hardness also holds for the general case without the above restrictions, this section contains some tractability results that hold even for an arbitrary alphabet. For instance, we prove the existence of problem kernels and fixed-parameter algorithms for

the parameter sought solution size in combination with the the alphabet size and the bound on the pairwise Hamming distance.

Nevertheless, with respect to the sought solution size k alone, the problem becomes intractable for an alphabet of unbounded size and so we finish the section with a proof of W[2]-hardness of Distinct Vectors with respect to k.

4.2.1 Problem Kernels

We start with a simple problem kernel with respect to the combined parameter (k, σ).

Proposition 4.6. *There exists an $O(\sigma^{\sigma^k+k})$-size problem kernel for* Distinct Vectors *computable in $O(d^2n)$ time, assuming constant-time arithmetical operations.*

Proof. From the fact that S contains n data points, it follows that at least $\lceil \log_\sigma n \rceil$ dimensions are required to distinguish all points. With fewer dimensions we could simply decide the instance with "no". Thus, we have $n \leq \sigma^k$.

Moreover, we claim that the number of dimensions d can always be bounded from above by σ^n. This bound is based on a simple data reduction rule: For any two dimensions that distinguish the same pairs of points (we call such dimensions *redundant*), we can arbitrarily delete one of them without altering the solution to the problem. This rule clearly is correct since any optimal solution does not contain both dimensions at once. Exhaustive application of the above rule requires $O(d^2n)$ time. The resulting instance is free of any redundant dimensions and so it holds that each dimension uniquely partitions the dataset into at most σ non-empty subsets. Thus, there are at most as many dimensions as there are partitions of n points into at most σ subsets:

$$d \leq \sum_{i=1}^{\sigma} S(n,i) = \sum_{i=1}^{\sigma} \frac{1}{i!} \sum_{j=0}^{i} (-1)^{i-j} \binom{i}{j} j^n.$$

Here, $S(n,i)$ is the *Stirling number of the second kind*, that is the number of ways to partition n points into i non-empty subsets. We skip a thorough analysis of the asymptotical behavior of the above sum and state the simple bound σ^n for the value of d. More dimensions cannot exist because otherwise there would be two dimensions with identical values for all n points, which clearly makes them redundant. The overall number of entries in S is thus in $O(\sigma^{\sigma^k+k})$, which yields a problem kernel. □

A problem kernel of size $O(\sigma^{\sigma^k+k})$ shows that Distinct Vectors is fixed-parameter tractable with respect to (k, σ). Recall Theorem 4.1, which shows that Distinct Vectors is also EW[2]-hard with respect to (k, σ) and thus implies that there is no linear-size problem kernel for Distinct Vectors. Nonetheless, the kernel stated in Proposition 4.6 seems to be a rather poor estimation which brings up the question for better problem kernels of polynomial size or even of singly exponential size. Filling the gap might be an interesting task.

As Charikar et al. [CGK+00] mentioned, Distinct Vectors can be polynomial-time reduced to Set Cover. Since Set Cover is equivalent to Hitting Set it is

also reducible to Hitting Set. This observation allows us to state a problem kernel for Distinct Vectors with bounded pairwise Hamming distance between any pair of points by transferring a known kernelization for a special case of Hitting Set, called f-Hitting Set, to our problem.

f-Hitting Set
Input: A finite universe U, a collection $\mathcal{C}$ of subsets of U of size at most f, and a nonnegative integer k.
Question: Is there a subset $K \subseteq U$ with $|K| \leq k$ such that K contains at least one element from each subset in $\mathcal{C}$?

Here, the value f corresponds directly to the maximum pairwise Hamming distance between pairs any pair of points in the Distinct Vectors instance as we will see shortly. Niedermeier and Rossmanith [NR03] showed an $O(k^3)$-size kernel for 3-Hitting Set. An $O(k^f)$-size kernel for the general f-Hitting Set is described in the book by Flum and Grohe [FG06]. A kernel with a universe of size $O(k^{f-1})$ was shown by Abu-Khzam [Abu10]. Van Bevern [Bev12] showed how a kernel comprising $f! \cdot f^{f+1} \cdot (k+1)^f$ subsets can be computed in $O(f|U| + f \log f \cdot |\mathcal{C}| + f|\mathcal{C}|^2)$ time. We use this kernelization to prove the following theorem:

Theorem 4.7. *There exists an $O((f! \cdot f^{f+1} \cdot (k+1)^f)^2)$-size problem kernel for* Distinct Vectors *computable in $O(n^2(d + f \log f + \log n) + f(n^4 + d))$ time, assuming constant-time arithmetical operations.*

Proof. The idea of the proof is to first describe a polynomial-time parameterized many-one reduction from Distinct Vectors to f-Hitting Set. Then one applies the aforementioned kernelization which yields an f-Hitting Set instance of size $O(f! \cdot f^{f+1} \cdot (k+1)^f)$. This kernel will then be transformed back by the reduction from the proof of Theorem 4.1. The resulting Distinct Vectors instance will have size $O((f! \cdot f^{f+1} \cdot (k+1)^f)^2)$, which yields the problem kernel.

The first reduction works as follows: Given an instance (S, k) of Distinct Vectors, the f-Hitting Set instance $(U, \mathcal{C}, k)$ is defined by

$$\begin{aligned} U &:= \{1, \ldots, d\}, \\ \mathcal{C} &:= \{C_{ij} \subseteq U \mid 1 \leq i < j \leq n\}, \\ C_{ij} &:= \{u \in U \mid (x_i)_u \neq (x_j)_u\}. \end{aligned}$$

Note that $|C_{ij}| \leq f$ for all $i \neq j$. This reduction requires $O(n^2 d)$ time. It is correct since $K \subseteq \{1, \ldots, d\}$ with $|K| \leq k$ is a solution of (S, k) if and only if for every pair of points in S there is at least one dimension in K in which both points have different values. This is equivalent to the situation that K contains at least one element from each C_{ij} in $\mathcal{C}$, which implies that K is a solution of $(U, \mathcal{C}, k)$.

Now we can apply the kernelization mentioned above to $(U, \mathcal{C}, k)$. In $O(fd + f \log f \cdot n^2 + fn^4)$ time we obtain an instance $(U', \mathcal{C}', k)$ of f-Hitting Set, where $\max\{|U'|, |\mathcal{C}'|\} \leq f! \cdot f^{f+1} \cdot (k+1)^f$.

Finally, we use the reduction from the proof of Theorem 4.1 to get the Distinct Vectors instance (S', k') in $O(n^2(d + \log n))$ time. Since $(U', \mathcal{C}', k)$ is an instance of f-Hitting Set, each point in S' is equal to 1 in at most f of the first $|U'|$ dimensions. Thus, each pair of points in S' differs in at most

$$f' = 2f + \log|\mathcal{C}'| = 2f + \sum_{j=1}^{f} \log j + (f+1)\log f + f\log(k+1)$$

dimensions. The new sought solution size is

$$k' = k + \log|\mathcal{C}'| = k + \sum_{j=1}^{f} \log j + (f+1)\log f + f\log(k+1).$$

Note that k' and f' depend only on k and f, which also holds for the overall size of S', which is in $O((f! \cdot f^{f+1} \cdot (k+1)^f)^2)$. The overall running time is in $O(n^2(d + f\log f + \log n) + f(n^4 + d))$. Thus we have a problem kernel. □

4.2.2 Fixed-Parameter Tractability and Approximation

The case of a bounded pairwise Hamming distance brings further tractability results with it. In addition to the problem kernel from Theorem 4.7, we give a fixed-parameter algorithm with respect to the combined parameter (k, f).

Proposition 4.8. Distinct Vectors *is solvable in $O(f^k \cdot nd)$ time, where f is the maximum pairwise Hamming distance and k is the requested solution size, assuming constant-time arithmetical operations.*

Proof. The following simple search tree algorithm solves a given Distinct Vectors instance (S, k) in $O(f^k \cdot nd)$ time: We set $K = \emptyset$ to start with. As long as $S_{|K}$ contains a pair of vectors x, y that cannot be distinguished yet, we determine the set

$$D := \{i \in \{1, \ldots, d\} \setminus K \mid (x)_i \neq (y)_i\}$$

of dimensions where x and y differ. Any solution has to contain at least one element of D and so we simply branch over all dimensions in D. Since x and y differ in at most f dimensions the resulting search tree is of size $O(f^k)$. Determining x, y and D can be done by lexicographical sorting the data via radix sort in $O(nd)$ time [Knu98] and comparing successive points afterwards. □

Albeit Distinct Vectors is NP-hard even for a constant $f = 4$ (see Theorem 4.4), there exists a simple factor-f approximation in $O(n^2 d)$ time.

Proposition 4.9. Distinct Vectors *can be approximated within a factor of f in time $O(n^2 d)$, assuming constant-time arithmetical operations.*

Proof. The algorithm is based on the following greedy strategy: Start with the empty set $K = \emptyset$. Consider a pair of points $x, y \in S_{|K}$ that cannot be distinguished yet. There are at most f dimensions in which x and y differ. We simply add all f dimensions to K and repeat this procedure until all points in $S_{|K}$ are pairwise distinct. Since any optimal solution has to contain at least one dimension in which x and y differ, our solution is at most f times larger than optimal. This can be done in time $O(n^2 d)$. □

4.2.3 W[2]-Hardness Regarding the Required Solution Size

As we have already seen in Corollary 4.5, the binary version of Distinct Vectors parameterized by the number of dimensions to be deleted is W[1]-hard. Moreover, the problem kernel given in Proposition 4.6 shows that Distinct Vectors is fixed-parameter tractable with respect to k for constant σ. Recall the reduction from the proof of Theorem 4.1 for which we already mentioned that it is not a parameterized one since the sought solution size k' does not solely depend on k but also on n. For this reason, the proof does not provide evidence of W[2]-hardness of Distinct Vectors for the binary case even though Hitting Set is known to be W[2]-hard [DF99]. To round off this section, we now show that Distinct Vectors is W[2]-hard with respect to the parameter k in case of an alphabet of unbounded size.

Theorem 4.10. Distinct Vectors *is* W[2]*-hard with respect to the parameter* k *for an unbounded alphabet* Σ*.*

Proof. We give a similar reduction from Hitting Set as in the proof of Theorem 4.1. An alphabet of unbounded size allows us to get rid of the last $\lceil \log n \rceil$ dimensions. As a result, we obtain a parameterized reduction which yields the W[2]-hardness.

We reduce a given Hitting Set instance $(U, \mathcal{C}, k)$ with $U = \{u_1, \ldots, u_m\}$ and $\mathcal{C} = \{C_1, \ldots, C_n\}$ to the Distinct Vectors instance (S, k'), where

$$S := \{x_1, \ldots, x_n, \vec{0}\} \subseteq \Sigma^m, \quad (x_i)_j := \begin{cases} i & u_j \in C_i \\ 0 & u_j \notin C_i \end{cases}, \quad k' := k$$

for all $i \in \{1, \ldots, n\}$ and $j \in \{1 \ldots, m\}$.

Let K be a subset of dimensions of size k that is a solution of (S, k'). Then, for every $i \in \{1, \ldots, n\}$, there exists a dimensions in K such that x_i is different from 0 in this dimension because $x_{i|K} \neq \vec{0}_{|K}$. It follows that the elements of U corresponding to the dimensions in K form a solution of $(U, \mathcal{C}, k)$.

Now, suppose that K is a solution of $(U, \mathcal{C}, k)$. The corresponding set of dimensions is a solution of (S, k') since, for every $i \in \{1, \ldots, n\}$, there is a dimension where x_i is equal to i. This implies that x_i can be distinguished from all other points because no other point contains the symbol i.

Thus, the reduction is correct and clearly runs in polynomial time. Note that the sought solution size k' only depends on the parameter k. Consequently, the above reduction is indeed a parameterized reduction. □

4.3 Summary

This section briefly summarizes the discussion of the Distinct Vectors problem. In Theorem 4.4, we have seen that it is NP-hard to solve even for very restricted instances where the alphabet contains two symbols and each pair of points differs in at most four dimensions. This variant is also W[1]-hard with respect to the number of dimensions to delete. Only if each pair differs in at most three dimensions, then the dataset exhibits a regular structure that allows for a polynomial-time algorithm to solve it as was shown in Theorem 4.3.

In general, if the size of the alphabet is unbounded, the problem even withstands approaches to fixed-parameter algorithms regarding the sought solution size since Theorem 4.10 shows that it is W[2]-hard. But, taking into account the size of the alphabet or the maximum Hamming distance between any pair of points together with the requested solution size, enables fixed-parameter algorithms as was shown in Proposition 4.6, Theorem 4.7 and Proposition 4.8

5 Conclusion

In this thesis, we studied several combinatorial feature selection problems. First, we considered the clustering problems Hidden Clusters and Hidden Cluster Graph. We have seen that both problems are NP-hard in general and so we investigated their parameterized complexity. The Hidden Clusters problem turned out to be W[2]-hard with respect to number of dimensions to select and fixed-parameter tractable with respect to the combination of the number of dimensions to be deleted and the number of cluster centers. For the Hidden Cluster Graph problem, where the number of cluster centers is unknown, we proved that it is W[2]-hard with respect to the number of dimensions to be deleted. Combining the number of dimensions to be deleted with the radius, however, yields a fixed-parameter tractable parameterization of Hidden Cluster Graph.

Besides the two clustering problems, we also studied the dimension reduction problem Distinct Vectors. We first focused on a very restricted case of Distinct Vectors, where we assumed a binary alphabet and a bounded pairwise Hamming distance of the data points. We recognized a dichotomous behavior in the computational complexity concerning the bound on the pairwise Hamming distance: If the bound is at most three, then Distinct Vectors is polynomial-time solvable. Otherwise it is NP-hard and even W[1]-hard with respect to the number of dimensions to delete. For alphabets of arbitrary cardinality, the problem is W[2]-hard with respect to the number dimensions to select. The problem, however, is fixed-parameter tractable with respect to the combined parameter number of dimensions to select and alphabet size. But there is in fact a lower bound on the running time since we proved the problem to be EW[2]-hard with respect to the above parameterization. Thus, Distinct Vectors parameterized by the number of dimensions to select and the alphabet size is in FPT $\setminus$ EPT. Also the bound on the pairwise Hamming distance in combination with the number of dimensions to select yields fixed-parameter tractability.

Future Work. Finally, we mention a few interesting questions that had to be left open in this thesis and thus offer some possibilities for further research. For instance, our discussion of Hidden Clusters is based solely on the special case Binary Hidden Clusters. It is thus natural to ask for fixed-parameter results concerning the general case of arbitrary alphabets and an arbitrary radius.

The Hidden Cluster Graph problem could be analyzed in respect of fixed-parameter tractability for a finite alphabet since our proof of W[2]-hardness required an unbounded alphabet. So far, it is also not clear whether the problem is fixed-parameter tractable with respect to the number of dimensions to select or not. Moreover, one could define several generalized versions of Hidden Cluster Graph. For example,

recall that we used the triangle inequality to prove a fixed-parameter algorithm for Hidden Cluster Graph with respect to the radius and the number of dimensions to be deleted. It could thus be interesting to consider arbitrary distance functions, for which the triangle inequality does not hold. It is also possible to modify the definition of a clustering. For example, one could define a relaxed notion of a cluster graph that allows for some edges between nodes of different clusters and some missing edges inside a cluster.

As regards the Distinct Vectors problem, the most interesting question is whether the problem kernel with respect to the number of dimensions to select and the alphabet size can be improved. The gap between $O(\sigma^{\sigma^k+k})$ and the linear lower bound leaves plenty of room for improvements. Note that a polynomial-size problem kernel would yield a fixed-parameter tractable algorithm that performs "as best as possible" because the EW[2]-hardness implies that one could not expect to do essentially better. Instead of improving the size of the problem kernel, one could also try to find better lower bounds on the size. For example, one could try to show that there is in fact no polynomial-size kernel using the cross-composition technique by Bodlaender et al. [BJK11]. It is also conceivable to find better problem kernels for alphabets of constant size. Another interesting point about Distinct Vectors is its relation to Hitting Set. We have seen that Distinct Vectors is essentially a special instance of a Hitting Set problem. For an alphabet of constant size, however, Distinct Vectors is fixed-parameter tractable with respect to the parameter sought solution size (that is, the number of dimensions to select), which is in contrast to Hitting Set generally being W[2]-hard with respect to the sought solution size. Thus, the Hitting Set instances corresponding to Distinct Vectors instances have to contain a certain structure that makes them easier to solve. Identifying and analyzing the structure of the corresponding Hitting Set instances is an interesting task that might reveal further parameterized complexity results.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. Cited on page 11.

[Abu10] Faisal N. Abu-Khzam. A kernelization algorithm for d-hitting set. *Journal of Computer and System Sciences*, 76(7):524–531, 2010. Cited on page 40.

[BCH+12] Robert Bredereck, Jiehua Chen, Sepp Hartung, Stefan Kratsch, Rolf Niedermeier, and Ondřej Suchý. A multivariate complexity analysis of lobbying in multiple referenda. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1292–1298, 2012. Cited on pages 24 and 28.

[Bel61] Richard Bellman. *Adaptive Control Processes - A Guided Tour*. Princeton University Press, 1961. Cited on page 1.

[Bev12] René van Bevern. Towards optimal and expressive kernelization for d-hitting set. Manuscript, 2012. Abstract appeared in Proceedings of the 18th Annual International Computing and Combinatorics Conference. Cited on page 40.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. Cited on page 8.

[BJK11] Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science*, pages 165–176, 2011. Cited on page 46.

[BL97] Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271, 1997. Cited on page 1.

[BM11] Andreas Brandstädt and Raffaele Mosca. On distance-3 matchings and induced matchings. *Discrete Applied Mathematics*, 159(7):509–520, 2011. Cited on page 38.

[Bod09] Hans Bodlaender. Kernelization: New upper and lower bound techniques. In Jianer Chen and Fedor Fomin, editors, *Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science*, pages 17–37. Springer, 2009. Cited on page 13.

[Cam89] Kathie Cameron. Induced matchings. *Discrete Applied Mathematics*, 24(1–3):97–102, 1989. Cited on page 37.

[CCDF97] Liming Cai, Jianer Chen, Rodney G. Downey, and Michael R. Fellows. Advice classes of parameterized tractability. *Annals of Pure and Applied Logic*, 84(1):119–138, 1997. Cited on page 12.

[CFRS07] Robin Christian, Mike Fellows, Frances Rosamond, and Arkadii Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007. Cited on pages 24 and 25.

[CGK+00] Moses Charikar, Venkatesan Guruswami, Ravi Kumar, Sridhar Rajagopalan, and Amit Sahai. Combinatorial feature selection problems. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 631–640, 2000. Cited on pages iii, v, 2, 3, 6, 7, 9, 17, 18, 31, 32, and 39.

[CM03] Carlos Cotta and Pablo Moscato. The k-feature set problem is W[2]-complete. *Journal of Computer and System Sciences*, 67(4):686–690, 2003. Cited on page 9.

[CM05] Carlos Cotta and Pablo Moscato. The parameterized complexity of multiparent recombination. In *Proceedings of the Sixth Metaheuristics International Conference*, pages 237–242, 2005. Cited on page 9.

[Das97] Manoranjan Dash. Feature selection via set cover. In *Proceedings of the IEEE Workshop on Knowledge and Data Exchange*, pages 165–171, 1997. Cited on page 9.

[DF99] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer-Verlag New York, 1999. Cited on pages 2, 12, 13, 20, 34, and 42.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, 2nd edition, 2001. Cited on page 8.

[Die10] Reinhard Diestel. *Graph Theory*. Springer, 4th edition, 2010. Cited on page 14.

[DR94] Scott Davies and Stuart Russell. NP-completeness of searches for smallest possible feature sets. In *AAAI Symposium on Intelligent Relevance*, pages 37–39, 1994. Cited on page 9.

[FG06] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006. Cited on pages 2, 12, 13, 14, 33, and 40.

[FGW06] Jörg Flum, Martin Grohe, and Mark Weyer. Bounded fixed-parameter tractability and $\log^2 n$ nondeterministic bits. *Journal of Computer and System Sciences*, 72(1):34–71, 2006. Cited on page 14.

[FH01] Karel Fuka and Rudolf Hanka. Feature set reduction for document classification problems. In *International Joint Conferences on Artificial Intelligence: Workshop on Text Learning: Beyond Supervision*, 2001. Cited on page 1.

[GE03] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003. Cited on page 8.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979. Cited on page 11.

[GWBV02] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1–3):389–422, 2002. Cited on page 1.

[HKSS12] Chính T. Hoàng, Marcin Kamiński, Joe Sawada, and R. Sritharan. Finding and listing induced paths and cycles. *Discrete Applied Mathematics* to appear, 2012. Cited on page 24.

[HM94] Kevin S. Van Horn and Tony Martinez. The minimum feature set problem. *Neural Networks*, 7(3):491–494, 1994. Cited on pages 1 and 9.

[Hof07] Marc A. Hoffmann. *Whisky: Marken aus der ganzen Welt*. Parragon, 2007. Cited on page 1.

[Jol02] I. T. Jolliffe. *Principle Component Analysis*. Springer, 2nd edition, 2002. Cited on page 2.

[Juk11] Stasys Jukna. *Extremal Combinatorics*. Springer, 2nd edition, 2011. Cited on page 34.

[Knu98] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison-Wesley, 2nd edition, 1998. Cited on pages 22, 37, and 41.

[Krz87] W. J. Krzanowski. Selection of variables to preserve multivariate data structure, using principal components. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 36(1):22–33, 1987. Cited on page 2.

[LM98a] Huan Liu and Hiroshi Motoda, editors. *Feature Extraction, Construction and Selection: A Data Mining Perspective*, volume 453 of *The Springer International Series in Engineering and Computer Science*. Springer, 1998. Cited on page 1.

[LM98b] Huan Liu and Hiroshi Motoda. *Feature Selection for Knowledge Discovery and Data Mining*, volume 454 of *The Springer International Series in Engineering and Computer Science*. Springer, 1998. Cited on page 1.

[LM07] Huan Liu and Hiroshi Motoda, editors. *Computational Methods of Feature Selection*. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC, 2007. Cited on page 1.

[LMS12] Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Kernelization – preprocessing with a guarantee. In Hans Bodlaender, Rodney G. Downey, Fedor Fomin, and Dániel Marx, editors, *The Multivariate Algorithmic Revolution and Beyond*, volume 7370 of *Lecture Notes in Computer Science*, pages 129–161. Springer, 2012. Cited on page 13.

[MBN02] Luis Carlos Molina, Lluís Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of IEEE International Conference on Data Mining*, pages 306–313, 2002. Cited on page 8.

[MT09] Hannes Moser and Dimitrios M. Thilikos. Parameterized complexity of finding regular induced subgraphs. *Journal of Discrete Algorithms*, 7(2):181–190, 2009. Cited on page 38.

[Nie06] Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. Cited on pages 2 and 12.

[NR03] Rolf Niedermeier and Peter Rossmanith. An efficient fixed-parameter algorithm for 3-hitting set. *Journal of Discrete Algorithms*, 1(1):89–102, 2003. Cited on page 40.

[OSV92] Arlindo L. Oliveira and Alberto Sangiovanni-Vincentelli. Constructive induction using a non-greedy strategy for feature selection. In *Proceedings of Ninth International Conference on Machine Learning*, pages 355–360, 1992. Cited on page 9.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994. Cited on page 11.

[Pea01] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901. Cited on page 2.

[RS00] Sam Roweis and Lawrence Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. Cited on page 8.

[SSM98] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998. Cited on page 8.

[SST04] Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1–2):173–182, 2004. Cited on page 15.

[TSL00] Josh B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. Cited on page 8.

[WMC+00] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems 13*, pages 668–674, 2000. Cited on page 1.

[XK01] Eric P. Xing and Richard M. Karp. CLIFF: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. In *Proceedings of the Ninth International Conference on Intelligent Systems for Molecular Biology*, pages S306–S315, 2001. Cited on page 1.