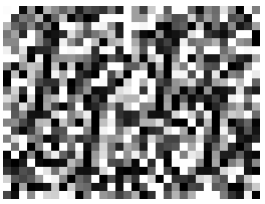
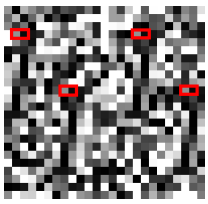
## 1 Introduction

In recent years, an emerging class of algorithms utilizing the *Minimum Description Length (MDL) principle* [**?**,**?**] have become more and more common in the field of explanatory data analysis. Examples of such approaches include the early Krimp [**?**] or the more recent Classy [**?**] algorithms. The MDL principle was first described by Rissanen in 1987 [**?**] as a practical implementation of Kolmogorov Complexity [**?**]. Central to MDL is the notion that 'learning' can be thought of as 'finding regularity' and that regularity itself is a property of data that is exploited by *compressing* said data. Therefore by compressing a dataset, we actually learn its structure — how regular it is, where this regularity occurs, what it looks like — at the same time. Indeed MDL postulates that the most optimal compression (minimal description) of a given dataset provides the best description of that data.

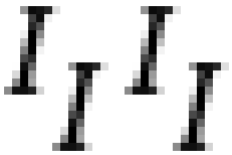
The problem that MDL tries to solve first and foremost, is that of *model selection*: given a multitude of explanations (models), select the one that fits the data best. In addition to this, MDL has also been demonstrated to be very effective in materialization of a specific model given the data. In this case, the model is predetermined and we want to find the parameters to fit the data. A similar problem class is solved in pattern mining: here the 'parameters' are the discrete building blocks that make up patterns in the data. In fact, the Krimp algorithm mentioned earlier solves a specific subclass of the problem. In this paper we will look at another class of pattern mining problems that is rarely addressed in literature, namely that of spatial structure discovery.



(a) $32 \times 24$ noise 'matrix'



(b) Pair $(146, 11)$



(c) Pattern 'I' occurs four times.

Fig. 1: Brief example of spatial pattern mining

Let us demonstrate the problem of spatial pattern mining by giving a brief example. Figure 1a shows a grayscale image that we assume is random 'white noise'. For convenience, we will interpret the image as a $32 \times 24$ matrix with values on the interval $[0; 255]$. If we were to look at all horizontal pairs of elements, we would find that the pair $(146, 11)$ is, among others, statistically more prevalent than the initial assumption of random would suggest. Figure 1b highlights the locations where these pairs occur: we have just discovered the first repeating structure in our dataset! If we would continue to try all combinations

of elements that 'stand out' from the background noise, Figure 1c shows that we will eventually find that the matrix contains four copies of the letter 'I' set in 16 point Garamond Italic.

The 35 elements that make up a single 'I' in the example, are said to form a *pattern*. We can use this pattern to describe the matrix in an other way than '768 unrelated values'. For example, we could describe it as 628 unrelated values plus pattern 'I' at locations $(5, 4), (11, 11), (20, 3), (25, 10)$, separating the structure from the accidental (noise) data. Since this requires less storage space than before, we have also compressed the original data. See how at the same time we have learned something about the data: we did not know about the 'hidden I's before.

### 1.1 Spatial pattern mining

As the example above very roughly demonstrates, spatial pattern mining is the problem of finding recurring (local) structure in multi-dimensional matrices of data. It is different from graph mining, as a matrix is more rigid and each element has a fixed degree of connectedness/adjacency. It is also unrelated to linear algebra, other then using the term 'matrix' and a comparable style of notation. Furthermore in this context, matrix elements can only be discrete, rows and columns have a fixed ordering and the semantics of a value is position-independent.

The problem of spatial pattern mining can roughly be divided into three classes. The first class consists of three subclasses: identical recurring structures in (1a) an otherwise empty (sparse) matrix, (1b) differently distributed noise and (1c) similarly distributed noise. The second class contains the same subclasses but adds that the recurring structures can also be overlaid with noise and are therefore not identical. The third class is also a continuation of the first class and requires that the recurring structures are identical after some optional transformation (such as mirror, inverse, rotate, etc.). These classes also represent an increasing difficulty level and serves as a rough benchmark for the performance of an algorithm.

Although we could intuitively solve the example, in reality it is much more complex. Is the 'I' the best pattern we can find (and why), are there any more patterns and, most important, how do we find it (quickly) in an unknown and/or much larger dataset? After formally defining these problems and their contexts we will introduce VOUW, a novel spatial pattern mining algorithm based on the MDL principle.

### 1.2 Preliminaries

Introduce the topics that will be discussed in this thesis.

- In this section we will introduce foundations such as Kolmogorov-complexity, MDL, entropy, etc.

- Furthermore we will give a brief description of the problem, with some examples
- The introduction is concluded with a subsection on related work
- In the section 'Theoretical Framework' we discuss the formal part of VOUW, with all of its definitions and a more complete description of the problem
- In the section 'A Search Algorithm' we will talk about the actual search algorithm and extensions thereof
- The (short) section 'Practical Implementation' will be about actual code and QVouw, the GUI
- 'Experiments' will try to empirically show some of the concepts introduced in the previous two sections
- And finally 'Discussion' and 'Conclusion' will conclude the thesis

### 1.3 General Problem Description

VOUW is a framework/set of definitions, a practical algorithm an academic implementation of these concepts. We use explanatory data mining to discover the structure of a dataset. To goals is not only to describe the data as concise as possible, but also how we derive the vocabulary to make that description. This concept can be extended from analysis to similarity and clustering: want we not only be able to tell how similar two datasets are, but also why they are (dis)similar.

**Brief Example** I would like to illustrate the problem with a small example, such that the reader's interest is (hopefully) piqued. This example could be a small fabricated matrix or (ideally) a toy example that bears some real-world relevance.

#### Grids, Images and Matrices

## 2 Introduction

The problem is defined on a specific type of input:

- Matrix-like data, but not in the linear algebra sense of the word. The closest description would be an image, because it also is a rigid, grid-like dataset where rows and columns have a fixed ordering.
- VOUW is, however, not an image mining algorithm and therefore we use the term 'matrix'.
- We use 2D input only. The algorithm and definitions could be expanded for multi-dimensional matrices.
- Individual data points must be discrete.
- Matrix can be sparse, but low-density matrices yield no useful patterns and thus the algorithm has limited usability on sparse matrices.

**Patterns** In order to gain better understanding of the input data, we try to express it in terms of a set of *patterns*. In this context a pattern is:

- A submatrix of the original matrix
- A structure that occurs more than once, either exact or with some degree of 'noise tolerance'

A complete, loss-less description does not only contain patterns, but also tells us where they are located in the original matrix. We call the placement of a pattern on the original matrix an *instance* of a pattern. A description of the original matrix is valid only if each element belongs to exactly one instance. There are, however, many valid descriptions possible. The third section discusses a search algorithm that tries to approach the optimal.

**The Problem and Its Classes** Apart from a formal definition, it is useful to semantically define what kinds of problems we would like to be able to solve. Therefore a division in five different classes was made.

**Class 1a** Exact duplicates in a sparse matrix.
**Class 1b** Exact duplicates in (differently distributed) noise.
**Class 1c** Exact duplicates in noise with equal distribution.
**Class 2** Approximate duplicates
**Class 3** Transformed duplicates