Explainable Data Decompositions

Sebastian Dalleiger, Jilles Vreeken
CISPA Helmholtz Center for Information Security
{sebastian.dalleiger, jv}@cispa.de

Abstract

Our goal is to discover the components of a dataset, characterize *why* we deem these components, explain *how* these components are different from each other, as well as identify what properties they *share* among each other. As is usual, we consider regions in the data to be components if they show significantly different distributions. What is not usual, however, is that we parameterize these distributions with patterns that are informative for one or more components. We do so because these patterns allow us to characterize what is going on in our data as well as explain our decomposition.

We define the problem in terms of a regularized maximum likelihood, in which we use the Maximum Entropy principle to model each data component with a set of patterns. As the search space is large and unstructured, we propose the deterministic DISC algorithm to efficiently discover high-quality decompositions via an alternating optimization approach. Empirical evaluation on synthetic and real-world data shows that DISC efficiently discovers meaningful components and accurately characterises these in easily understandable terms.

Introduction

Suppose we are analysing the sales data from a supermarket. Likely, our clientele consists of different groups of customers, each of which have their own buying behaviour. Students, for example, often buy *pasta* and *ketchup*, post-docs buy ready-made *sauce* with their *pasta*, whereas professors can afford to make the sauce themselves out of fresh ingredients. That is, the data consists of different *components*, parts of the data that show significantly different pattern distributions.

Certain patterns may be characteristic for more than just one component; both students and post-docs often consume *pizza* and *beer*, after all. That is, the set of patterns that characterise the data can also be partitioned: each such *pattern component* consists of those patterns that are characteristic for a distinct set of *data components*. Together, the pattern components give detailed yet easily interpretable insight in why there are components, how they are different from each other, and what properties are shared among them.

We propose to discover the *pattern composition* of a given database. That is, our goal is to jointly discover the components of the data as well as those pattern components that optimally characterise their similarities and differences—and do so both efficiently and in a statistically well-founded manner where we only have to set a significance threshold α. To this end, we model a component given a set of patterns using the Maximum Entropy principle (Jaynes 1982). That is, we use a maximum likelihood estimator that satisfies the empirically observed frequencies of the given patterns, but otherwise makes no further assumptions. We can then formulate the problem in terms of a likelihood maximization problem, where we after that composition that achieves the highest overall likelihood. To avoid overfitting we rely on the BIC model selection criterion. In other words, we are after the most succinct way to summarise the data, by partitioning, such that parts exhibit significantly different distributions, and describing these distributions non-redundantly using only small and interpretable pattern sets.

Clearly, the search space for this problem is enormous: for a given dataset there exist an exponential number of patterns, an exponential number of pattern sets, and an exponential number of partitions. Moreover, this space is not structured, barring efficient search for the optimum. We therefore introduce DISC, a deterministic method that heuristically discovers a good pattern composition. The main idea is that we split the problem into two parts, and iterate between them until convergence. That is, for a given data decomposition we propose to approximate the pattern components using DESC, and then given those pattern components to DISC to discover refined data components. In both steps we rely on statistical tests to both prune the search space, as well as make sure we do not discover any spurious patterns or components.

Extensive experiments show that DESC and DISC work well in practice. DESC outperforms the state of the art in pattern set mining, highly efficiently discovering succinct models, while DISC recovers meaningful components. Case studies confirm that the components and their characterisations make sense: the ecological niches and commonalities that DISC discovers correspond to the ground truth.

In sum, the contributions of this paper are as follows, **(I)** defining the *pattern composition* problem, **(II)** a fast method for discovering pattern components, **(III)** a fast method for discovering pattern compositions, and **(IV)** extensive validation on synthetic and real data. We provide the proofs, additional experiments, and details for reproducibility in the supplementary material.

Preliminaries

In this work, we consider binary transaction data. Let $\mathcal{I}$ be a set of items, e.g. products for sale in a supermarket. A transaction $t \in \Omega$ is a set of items, e.g. the products a customer bought, where $\Omega = 2^{\mathcal{I}}$ denotes the space of all possible transactions over $\mathcal{I}$. A dataset D over $\mathcal{I}$ is then a bag of n transactions, e.g. the sales transactions on a given day. We write $\pi(D) \in \Pi(D)$ for a partitioning of data D into k non-empty subsets, $\pi(D) = \{D_1, \ldots, D_k\}$ s.t. $\bigcup_{D_j \in \pi(D)} D_j = D$. Whenever D is clear from context, we simply write π.

As patterns we are interested in itemsets $x \subseteq \mathcal{I}$. A transaction $t \in D$ supports an itemset $x \subseteq \mathcal{I}$ iff $x \subseteq t$. The empirical frequency of an itemset x in D is denoted by $q(x \mid D) = \frac{1}{n} |\{t \in D \mid x \subseteq t\}|$, using the shorthands $q(x)$ and $q_j(x)$ to denote the empirical frequency of x in D resp. D_j. A *pattern set* S is simply a set of itemsets. Combined with empirical frequencies, a set of patterns S defines a probability distribution p over Ω. For any $n \in \mathbb{N}$ we write $[n] = \{1, 2, \ldots, n\}$. All logarithms are to base 2, and by convention we use $0 \log 0 = 0$.

The Pattern Composition Problem

In this section we define our problem. Starting informally, our goal is to characterize the composition of a given binary dataset D. That is, we aim to decompose the dataset D into disjoint components $D_1, \ldots, D_k$, such that every component D_j has a significantly different pattern distribution p_j, while at the same time characterizing each of these distributions in terms of informative patterns, such that we can identify the similarities and differences between the components.

In other words, we our goal is to discover a partitioning π of D, a succinct, non-redundant set of patterns S, and an assignment A that associates patterns to components, such that we maximize the regularized likelihood

$$\ell(\pi, S, A) = -\sum_{D_j \in \pi(D)} \log p(D_j \mid S_j) + r(\pi, S, A) \; .$$

where $S_j \subseteq S$ is the subset of S that A indicates to be relevant for D_j. Last, $r(\pi, S, A)$ is a regularization term that steers the problem away from trivial solutions, such as decomposing the data into singleton components, or to including every possible pattern.

We discuss and define each of these terms in detail below, after which we give a formal definition of the problem.

The Probability Distribution

We start by defining the probability distribution p over Ω. It is easy to see that maximum likelihood estimator (MLE) of p depends on the patterns in S and their empirical frequencies in D. That is, p will only achieve maximum likelihood if for any pattern $x \in S$ the estimated frequency $p(x \mid S) = \mathrm{E}_f\left[x \mid S\right]$ corresponds to the observed frequency $q(x \mid D)$. We define the set of feasible distributions as the polytope

$$\mathbb{P}_S^D \equiv \{f \in \Omega \to [0,1] \,|\, \mathrm{E}_f\left[x \mid S\right] = q(x \mid D) \; \forall x \in S\} \; ,$$

that contains all, infinitely many distributions consistent with the empirical observations q of patterns $x \in S$ in D.

Not all distributions in $\mathbb{P}_S^D$ suit our needs: we need a distribution that does not introduce additional assumptions beyond the information that S specifies. From an information theoretic point of view, additional assumptions correspond to additional information. We can measure the amount of information in a distribution using Shannon entropy, $H(p) = -\sum_x p(x) \log p(x)$. The lower the information content of a distribution p, the higher its entropy. We can uniquely identify the feasible distribution that makes the least additional assumptions as the one with the highest entropy

$$f \equiv \underset{f \in \mathbb{P}}{\arg\max} \; H(f) \; ,$$

which is known as the Principle of Maximum Entropy (Jaynes 1982). In general this does not immediately provide a family of distributions to use. In our case, however, as the constraints of $\mathbb{P}_S^D$ are linear, we know that distribution p over transactions $t \in \Omega$ takes an exponential form

$$f(t \mid S) = \theta_0 \prod_{x_i \in S} \theta_i^{\mathbf{1}[x_i \subseteq t]} \; ,$$

that for appropriately chosen coefficients $\theta \in \mathbb{R}^{|S|+1}$ factorizes into marginals (Csiszár 1975). Conveniently, optimizing θ is a convex problem, and hence we can employ standard convex optimizers such as iterative scaling (Darroch and Ratcliff 1972). We are specifically interested in inferring the expected frequency $p(x \mid S)$ for arbitrary itemsets $x \in \Omega$—the frequencies of $x \in S$ are given, after all. To infer p we have to sum the probabilities of *every possible transaction* $t \in \Omega$ that supports x,

$$p(x \mid S) = \mathrm{E}_f\left[x \mid S\right] = \sum_{t \in \Omega} f(t \mid S) \, \mathbf{1}\left[x \subseteq t\right] \; ,$$

which is known to be **PP**-hard (Tatti 2006).

Usually, the assumption is that the full dataset is generated from *i.i.d.* samples, however, a component $D_j \in \pi(D)$ is defined to be a subset of D with a unique distribution. Within a component, samples are independently drawn. The likelihood of D is the product of component likelihoods, given sets S_j, where the likelihood of D_j is

$$p(D_j \mid S_j) = \prod_{t \in D_j} p(t \mid S_j) \; .$$

Wherever clear from context, we will write p_S for $p(\cdot \mid S)$, and p_j for $p(\cdot \mid S_j)$.

Informative Patterns

Our goal is to discover that maximally succinct, maximally non-redundant pattern set S that maximizes the likelihood of the data, $p_S(D)$. Likelihood ℓ is a monotonic function: if we insert any pattern $x \notin S$ into S it almost always increases, and at worst stays the same.

We say that a pattern x is *informative* for D with respect to S if we see a *significant* increase in likelihood if we include x in S. To determine whether a pattern or component is informative, we use a model selection criterion. We use BIC, as it is simple, efficiently computable, and as we will see, works well in practice (Schwarz 1978). For a single component, we have $|S| = m$ degrees-of-freedom (dof) and hence $r(S) = m/2 \log |D|$.

It is straightforward to generalize the above to multiple components. Given a partitioning π and a pattern set S, we need to determine which patterns $x \in S$ are informative for which component $D_i \in \pi$. This is what our assignment matrix $A \in \{0,1\}^{m \times k}$ is for. It is a binary matrix over components and patterns where $A_{ij} = 1$ if pattern $x_i \in S$ is informative for component D_j. The set of patterns that are informative for a component D_j is defined as $S_j = \{x_i \in S \mid A_{ij} = 1\}$. If a pattern is informative for multiple components, we call it common or shared among those components.

Taking assignment matrix A into account, the BIC regularization cost r is

$$r(\pi, S, A) = \frac{1}{2}[k \cdot (2m + d) + n] \log n ,$$

where **(i)** the assignment matrix for patterns is accounted for by having $m \times k$ dof. Next, **(ii)**, the number of coefficients, θ used by k distributions amounts to $(m + d) \times k$ additional dof, where $d = \dim D$ correspond to the additional dof used for singletons from $\mathcal{I}$. Thirdly, **(iii)** we have additional $|D|$ dof that can encode any partitioning, by assigning a label to each row. However, the latter is constant for any partitioning.

The Problem, formally

Combining the above, we can now formally state the problem.

Problem 1 (The Pattern Composition Problem) *Given a transactional dataset D over items $\mathcal{I}$, our goal is to jointly discover that **(i)** partitioning $\pi \in \Pi(D)$ of D into fewest parts, that **(ii)** smallest pattern set $S \subseteq T_{\mathcal{I}}$, and that **(iii)** assignment matrix $A \in \{0,1\}^{|S| \times |\pi|}$ such that*

$$\ell(\pi, S, A) = - \sum_{D_j \in \pi(D)} \log p(D_j \mid S_j) + r(\pi, S, A)$$

is minimal.

Unsurprisingly, this is a difficult problem with a very large search space. First of all, there exist a Bell-number $B_{|D|}$ of possible partitionings π. Secondly, the number of possible pattern sets is doubly exponential in the number of unique items in D, as $S \in 2^{2^{\mathcal{I}}}$. Finally, the objective function does not exhibit any structure that we can exploit for efficient search, i.e. it is neither (anti-)monotone, nor submodular.

Algorithms

To efficiently discover good solutions in practice, we separate the problem into two parts and take an alternating optimization approach. That is, starting from partitioning π_0 where all of D is in one part, we iterate between the following two steps until convergence. First, given a partitioning $\pi(D)$ we efficiently discover a high quality pattern set S and assignment matrix A. Second, given a pattern set S, assignment matrix A, and partitioning π we discover a refined partitioning π' such that we minimize our objective function.

Below we discuss these two steps in turn.

Discovering Patterns given a Partitioning

The first problem we consider is that of discovering a high-quality set of informative patterns $S \in 2^{\Omega}$ and assignment matrix A for a given partitioning $\pi \in \Pi(D)$. Like our overarching problem, this problem is also too hard to solve exactly; there exist doubly exponentially many pattern sets, and our hard-to-compute score does not show any structure we can exploit. We are hence going to approximate the optimal result through an iterative greedy approach.

For a single dataset, Mampaey, Vreeken, and Tatti (2012) showed that finding the set S with minimal ℓ is equivalent to finding the set that has the highest gain in likelihood in comparison to the empty pattern set $S_\emptyset$. That is, minimizing

$$\ell_j(S) = - \sum_{t \in D_j} \log p_j(t) ,$$

is equivalent to maximizing the KL-divergence

$$\arg\max_S \; KL\,[\, p_S \parallel p_\emptyset \,] \tag{1}$$

and the greedy solution to Eq. (1) is equivalent to iteratively minimizing $\ell_j(S)$ directly. In fact, we can guarantee the quality of the greedy solution.

Lemma 1 *Eq.* (1) *is a* Submodular Function Maximization *problem. The greedy solution S is in the e^{-m/m^*}-radius of the optimal solution S^*, where $m = |S|$ and $m^* = |S^*|$*

For conciseness, we provide the proof in the supplementary.

The greedy algorithm to minimize Eq. (1) is still prohibitively slow in practice: it repeatedly evaluates KL-divergences $KL[\, p_S \parallel p_{S \cup \{x\}} \,]$ to measure the information gain of adding itemset x to S, and this computation relies on computationally costly inference of frequencies using p. Nonetheless, this formulation does allow us to derive a computationally efficient admissible heuristic.

To reduce the complexity of computing KL, we want to reduce the number of queries it makes to p. In its full computation, it both considers the frequencies of pattern x itself, as well as that of its exponentially many subsets $y \subset x$. Ignoring these subsets permits the following lower-bound

$$h(x \mid S) = q(x) \log q(x)/p_S(x) .$$

Due to the decomposition of the data, this admissible heuristic easily generalizes to an admissible normalized *lower-bound information gain* over multiple components $h(x \mid S)$ is $\mathrm{E}_{S_j}[h(x \mid S_j)]$. In general, our pattern set discovery strategy is hence as follows. In the current iteration i, the last pattern set S^{i-1} is known and fixed. We use h to select that x in candidate set $F \subseteq \Omega$ that has the highest marginal gain. That is, until convergence of ℓ, we iterate

$$S^i \leftarrow S^{i-1} \cup \arg\max_{x \in F} h(x \mid S^{i-1}) .$$

This leaves us to specify the candidate set F. Naively, we could set $F = \Omega$. Clearly, this is not practical: besides that Ω is typically prohibitively large, it contains exponentially many candidates that will be uninformative with regard to patterns in S. We hence propose a more effective search strategy, in which we take into account what S can already

Algorithm 1: DESC for Describing the Composition

```
   Input: Data $D$, partitioning $\pi$
   Output: Distributions $p_j$, pattern set $S$, assignment $A$
1  $S \leftarrow \{x \in \mathcal{I}\}$
2  $A \leftarrow$ initialize with all 1s
3  $p \leftarrow$ infer $p(\cdot \mid S_j)$ for each component $D_j$
4  $F \leftarrow \{z = x \cup y \mid x, y \in S, r(z) < h(z)\}$
5  while $F \neq \emptyset$ do
6      $z \leftarrow \arg\max_{x \in F} h(x \mid S)$
7      $A' \leftarrow$ according Eq. (2) wrt $z$
8      $S' \leftarrow S \cup \{z\}$ if $z$ assigned to a component
9      $p' \leftarrow$ infer $p(\cdot \mid S'_j)$ for each component $D_j$
10     if $\ell(\pi, S', A') < \ell_j(\pi, S, A)$ then
11         $A \leftarrow A'$; $S \leftarrow S'$; $p \leftarrow p'$
12         $F \leftarrow \{z = x \cup y \mid x, y \in S, r(z) < h(z \mid S)\}$
13     else
14         $F \leftarrow F \setminus z$
15 return $(p, S, A)$
```

explain well. In a nutshell, we iteratively generate candidates by merging pairs of patterns $x, y \in S \cup \mathcal{I}$ into a candidate $x \cup y \in F$. However, we only want to consider the subset of candidates that will surely reduce our objective ℓ. Those are candidates $z \in F$ for which $h(z \mid S) > r(z)$, where $r(z) = r(\pi, S \cup \{x\}) - r(\pi, S)$. Similarly, we assign a candidate i to a component j if it yields a gain in ℓ_j, i.e.

$$A_{ij} = 1 \iff h(x_i \mid S_j) < r_j(x_i) \ , \qquad (2)$$

where the cost $r_j(z)$ is $r(\pi, S, A') - r(\pi, S, A)$. Here A' is equivalent to A, but with $A_{ij} = 1$.

Putting the above together, we have algorithm DESC, for which we give the pseudo-code as Algorithm 1. In short, starting with the singleton only model (line 1–3) we generate our initial batch of candidates F (ln. 4). We consider these candidates descending on h (ln. 6) and evaluate each $z \in F$ (ln. 7–9). If the objective improves, we keep the candidate (11–12), and otherwise reject it (14).

The computational complexity of DESC depends on the number of candidates in F, which is quadratic in the number of patterns in S and in the worst case can grows up to $|\Omega|$. In practice, however, the properties of the maximum entropy distribution together with the BIC regularizer keep the size of S small, in the order of tens to hundreds of patterns, say $S_{\max}$. The worst case complexity of DESC is dominated by the inference of the distributions p_j, and hence in PP. The average complexity, γ, of p is much lower (Mampaey, Vreeken, and Tatti 2012), however, and hence the average complexity of DESC is $\mathcal{O}(\gamma \cdot |S_{\max}|^2)$.

Discovering the Composition

Next, we consider the orthogonal problem of discovering a high-quality partitioning $\pi(D)$ given a pattern set S and assignment matrix A. As there is no effective exact search for the optimal partitioning, we again rely on heuristics. In particular, we take a top-down approach where we iteratively refine the current partitioning π using the patterns in S.

Our strategy is based on the idea that a significantly different distributions of patterns are an indicator for the presence of latent factors of unknown components. In other words, we say that a component was generated using a latent data source that left a distinctly distributed trail of patterns in the data. By narrowing down a given component to a subset with a distribution that stands out from the rest of the data, we can refine the current partitioning to identify these latent parts as separate components. We can also use this observation also in reverse: when we narrow down a component and find that the pattern distribution we so obtain is not significantly different from the remainder or the other components, we do not want this candidate component to be part of our solution.

We write p_j^x for the pattern distribution we infer on that part of component D_j where pattern x occurs. Likewise, we consider $p_j^{\not x}$ over that part of D_j where x does not occur. We measure the divergence between two distributions with same support using the Jensen-Shannon divergence $JS(P, Q) = KL[P \parallel M] + KL[Q \parallel M]$, where $M = (P + Q)/2$. The scaled $JS(p_j^x, p_j^{\not x})$ statistics is asymptotically χ^2 distributed with $|S| - 1$ dof (Menéndez et al. 1997). From this we get a p-value for a single test. However, as we test many hypotheses, i.e. candidate refinements, we hence correct for the familywise error-rate (FWER) by adjusting the significance level α using Bonferroni correction (Bonferroni 1936).

For a given S and for any partitioning $\pi \in \Pi(D)$, we write $A(\pi, S)$ as the assignment matrix that characterizes the partitioning π with S by minimizes our objective function with respect to Eq. (2). Formally, for a given S, the problem of discovering components is as follows

$$\begin{aligned} &\underset{\pi \in \Pi(D)}{\arg\min} && \ell(\pi, S, A(\pi, S)) \\ &\text{subject to} && p_i, p_j \text{ significantly JS-divergent } \forall i \neq j \end{aligned}$$

This is, again, a hard problem, and again, the search space is large and unstructured. We therefore employ a greedy top-down approach. Starting with a single component $D_j \in \pi$, we decompose D_j into two sub-components D_j^1 and D_j^2 such that these are significantly differently distributed from each other, as well as from the rest of other components in π. Following the notion that latent factors are identifiable by distinct pattern distributions, we start the refinement process of a given component $D_j \in \pi$ with a pattern $x \in S \times \mathcal{I}$ by separating a component into two children $D_j^x \equiv \{t \in D_j \mid x \subseteq t\}$ and $D_j^{\not x} \equiv D_j \setminus D_j^x$. The corresponding refinement of π is written as

$$\text{refine}_\pi(x, j) \equiv \{\pi', A(\pi')\} \ ,$$

where the new partitioning π' is $\pi \setminus \{D_j\} \cup \{D_j^x, D_j^{\not x}\}$.

As real data is noisy and distributions are complex, it is unlikely that an individual pattern x perfectly identifies a latent component. That is, after splitting a component it may be that the overall assignment of transactions to components may be suboptimal with regard to likelihood. Just like in the EM algorithm we therefore iteratively reassign transactions that components where they achieve the highest likelihood. That is, in each iteration we ensure for every $t \in D$ that

$$t \in D_{\hat{\imath}} \iff \hat{\imath} = \underset{j \in [k]}{\arg\max}\, p(t \mid S_j) \ , \qquad (3)$$

Algorithm 2: DISC for Discovering the Composition

```
   Input: Data $D$, significance threshold $\alpha$
   Output: Partitioning $\pi$, pattern set $S$, assignment $A$
1  $\pi \leftarrow \{D\}$
2  $S, A \leftarrow \text{DESC}(D, \pi)$
3  $G \leftarrow$ according Eq. (4)
4  while $G \neq \emptyset$ and $\ell$ has not converged do
5      $S, A \leftarrow \text{DESC}(D, \pi)$
6      $c \leftarrow \arg\min_{c \in G} \ell(\text{refine}_\pi(c), S)$ cf. Eq. (5)
7      $\pi', A' \leftarrow \text{refine}_\pi(c)$
8      while $\ell$ has not converged do
9          let $\pi'$ satisfy Eq. (3)
10         $A' \leftarrow A(\pi')$ according to Eq. (2)
11         $p \leftarrow$ infer $p(\cdot \mid S_j)$ for each component $D_j$
12     if $\ell(\pi', S, A') < \ell(\pi, S, A)$ then
13         $\pi \leftarrow \pi'$, $A \leftarrow A'$
14         $G \leftarrow$ according Eq. (4)
15     else
16         $G \leftarrow G \setminus c$
17 return $(\pi, S, A)$
```

re-estimate the distribution p, re-compute $A(\pi, S)$ and repeat until convergence. Starting with $\pi = \{D\}$, we iteratively refine the current partitioning by selecting the JS-significance refinement of π with highest marginal gain, until convergence of ℓ. Formally, out of the set G of candidates

$$\{(j, x) \in [k] \times S_j \cup \mathcal{I} \mid \text{refine}_\pi(x, j) \text{ significant}\} \ , \qquad (4)$$

we select the refinement candidate that reduces ℓ most

$$\underset{x,j \in G}{\arg\min}\, \ell(\text{refine}_\pi(x, j), S) \ . \qquad (5)$$

Putting all the above together, we have the DISC algorithm. We give the pseudo-code as Algorithm 2. In a nutshell, starting from the trivial partitioning $\pi(D) = \{D\}$, we iteratively use DESC to discover the pattern components, use these patterns to find the best refinement of π, reassign the rows to optimize the likelihood, and only accept this refinement if it is significant. We repeat this until convergence. Disregarding the complexity of querying our distribution, DISC scales linearly with the size of G. In the worst case this means $|\Omega| \times |D|$. However, since in practice both S and π tend to be small, DISC is feasible on real world data.

Related Work

The vast majority of literature has been devoted on either finding clusters of transactions or finding patterns that characterise a dataset. Surprisingly, there exist no technique to discover the *pattern composition* of the data.

Our problem obviously relates to mixture modelling (Dempster, Laird, and Rubin 1977) where data is modelled as a mixture of several probability distributions. Mixture modelling, however, requires us to assume a probability distribution, whereas the true distribution is unknown. Similarly, clustering (MacQueen 1967) is related, as it groups data points, but relies on an assumed distance measure. Additionally in contrast to our approach, many of these approaches are stochastic, require us to choose the number of components up front, and none characterise the commonalities and differences between the components in interpretable terms.

In this sense, co-clustering, also known as bi-clustering, is more closely aligned to our goal. In co-clustering we simultaneously cluster rows and columns and can interpret the column-clustering as an implicit characterisation of the row-cluster. Moreover, there exist parameter-free methods like Information Co-clustering (Dhillon, Mallela, and Modha 2003), Cross-associations (Chakrabarti et al. 2004). These techniques, only discover non-overlapping rectangles in the data that are exceptionally dense or sparse, rather than data components with significantly different pattern distributions.

There also exist methods that can provide post-hoc explanations, for example using a consistent set of decision rules, which lead to a prediction (Lakkaraju, Bach, and Leskovec 2016) or a clustering (Kim, Shah, and Doshi-Velez 2015; Chen et al. 2016). These rules together characterise the decision boundary for a cluster, whereas we are interested those patterns that characterise the similarities and differences between components. In other words, rather than explaining the clustering after the fact, our models directly explain *why* there *is* a clustering.

Pattern mining methods are obviously strongly related to DISC. Frequent pattern mining (Agrawal and Srikant 1994; Nijssen, Guns, and De Raedt 2009) is well-known to discover far too many patterns for the result to be interpretable. OPUS (Webb and Vreeken 2014; Webb 2010) curbs the pattern explosion through the use of statistical tests. SLIM (Smets and Vreeken 2012), IIM (Fowkes and Sutton 2016) and MTV (Mampaey, Vreeken, and Tatti 2012) are examples of techniques that discover concise and non-redundant pattern sets. These methods all only provide a single pattern set for a single database. DIFFNORM (Budhathoki and Vreeken 2015) is the only method we know that for a given data partitioning can characterise differences and similarities between the pattern distributions.

Experiments

In the experiments, we evaluate DISC on synthetic, as well as 17 real-world datasets that together span a wide variety of domains, sizes, and dimensionalities. We implemented DISC in C++ , ran experiments on a 12-Core Intel Xeon E5-2643 CPU, and report wall-clock time. We provide the source code, datasets, synthetic dataset generator, and additional information needed for reproducibility.[1]

In many of the following experiments we compare likelihood ℓ of the estimated model with the likelihood $\ell^{\emptyset}$ of the initial model, that is $S^{\emptyset} = \mathcal{I}$ for a single component $\pi(D) = \{D\}$. We measure the likelihood ratio $\ell/\ell^{\emptyset}$, in percent, where a lower values corresponds to a higher regularized likelihood of the data under the model. In all experiments we have used the same significance level $\alpha = 0.01$.

[1]https://eda.mmci.uni-saarland.de/disc/

Describing Components In this section, we study our pattern set miner DESC on real-world datasets. Before we characterize datasets for a given partitioning, we start with the special case of discovering a pattern set for a given composition. In this set-up, we compare against SLIM (Smets and Vreeken 2012), MTV (Mampaey, Vreeken, and Tatti 2012) and DIFFNORM (Budhathoki and Vreeken 2015). For efficiency reasons they only consider frequent patterns, i.e. for which $q(z) > \kappa$ according to a user defined minimum frequency threshold κ. It is trivial to constraint DESC to consider frequent patters only and to compare fairly, we use the same thresholds for all methods in the following experiments.

DIFFNORM characterizes a pre-partitioning dataset $\pi(D)$. To this end we consider 9 labeled datasets, which we partition based on their class labels. Neither SLIM, MTV make use of any partitioning and can only be applied on the complete dataset, while DESC can do both and is applied to partitioned data where available and otherwise to the complete data.

In Fig. 1a we show that SLIM discovers pattern sets that consist of hundreds up to thousands of patterns, the results of MTV and DESC are in the order of tens of patterns. The pattern sets discovered by DISC, DESC and MTV are much more concise than the results from SLIM for a single component or DIFFNORM for a given composition.

We give the wall-clock runtime of the three methods in Fig. 1b. We see that DESC achieves these results in an order of magnitude faster than MTV and surpasses SLIM. On average DESC requires less than a quarter (16%) of the runtime of SLIM, just 13% of DIFFNORM and only 0.24% of the runtime of MTV. As MTV and DESC optimize the same score, we can fairly compare them. In Fig. 1c we show that DESC outperforms MTV in almost all cases.

Discovering the Composition Now, we study the full algorithm: simultaneously discover both the pattern sets and the partitioning of the dataset using DISC. First, we test and verify DISC on synthetic data with 128 items in $\mathcal{I}$. For this, we generate synthetic datasets such that we have access to the ground-truth. In each trial, we randomly sample a dataset D, containing $1, 2, 4, 8$ components. For each component $D_j \in \pi$ we randomly generate and insert 5 characteristic patterns into S_j. For any disjoint pair D_i, D_j, we generate 3 shared patterns with probability of 20%, that are inserted into both S_i and S_j. Every pattern has a randomly chosen frequency associated with it. Each component D_j consists of 256 rows. In each row, we uniformly at random insert each pattern from S_j with its corresponding frequency. Lastly, we introduce additive noise, by randomly insert items into each row independently with probability of 5%. For each k^*, we sample 20 datasets and compare the ground-truth with DISC and DESC. On average, DISC reaches a likelihood ℓ^{DISC} within 2% of the ground-truth, i.e. $\ell^{\text{DESC}_{\pi^*}} \pm 2\%$ and always recovers the ground-truth number of components k^*.

Now after we have verified that DISC works on synthetic examples, we study DISC on the real-world datasets. To do so, we measure how similar data within a component is, using ℓ, and measure how differently distributed the components are, using the *pairwise symmetric KL-divergence* (**PSKL**)

$$\frac{1}{2\binom{k}{2}} \sum_{ij \in \binom{[k]}{2}} KL\left[p_i \| p_j\right] + KL\left[p_j \| p_i\right] ,$$

which averages the divergence between pairs of distributions.

Next, we compare DISC to clustering. While options include k-means, and Expectation-Maximization, these are not a good fit for our case as they are stochastic, require a number of clusters, and neither is the concept of a centroid well defined for discrete spaces, nor is it clear what efficiently queryable distribution to use. DBSCAN (Ester et al. 1996) relies on a distance measure, that we define as

$$d(s,t) = 1 - |s \cap t| \,/\, \max(|s|, |t|) .$$

Our approach is as follows: First, we cluster the dataset using DBSCAN and get $\pi \in \Pi(D)$. Next, we use DESC to describe the clusters π post-hoc by means of S, p and $A(\pi, S)$. Since DBSCAN relies on hyper-parameter, we optimize ℓ using a grid-search over 7 ϵ-candidates and we do not constraint cluster-sizes. We call this algorithm DBDESC. Similarly, we define DBDESC2 that uses a different distance measure $d'(s,t) = d(c(s), c(t))$, where $c(t)$ contains patterns from S that are subsets of t, i.e. $\{s \in S \mid s \subseteq t\}$.

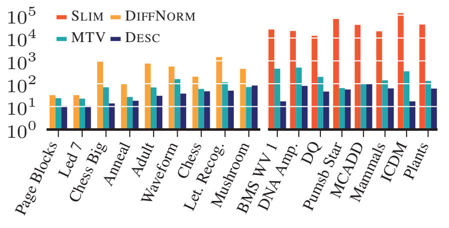
We apply DISC, DBDESC and DBDESC2 on all 17 datasets without using any class-labels and summarize results in Fig. 2. First of all, we compare the composition of DISC, DBDESC and DBDESC2 with a class-based composition discovered by DESC on 9 labeled datasets. Almost always, we observed a significantly lower PSKL-divergence between classes than between components, which we show in Fig. 2b. Noticeable is the PSKL-divergence of *Pumsb Star*, as with only 10 components, the divergence is very high compared to DBDESC and DBDESC2.

Additionally, DISCs model usually has a significantly lower objective function ℓ than the class-based result from DESC. This suggest that classes are not always good indicator for components. We see a higher likelihood for decomposed data in comparison to DESC without decomposing the data on class labels, in Fig. 2c. We note the much higher likelihood of the DISCs composition in comparison to DBDESC and DBDESC2. Overall we see that DISC discovers diverging components that have higher in likelihood in comparison to the cluster based composition from DBSCAN.

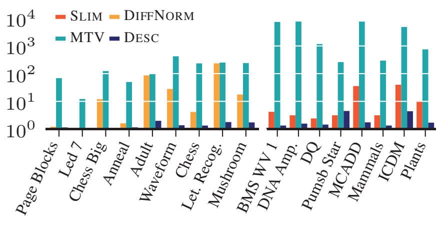
Qualitative Study Now, we study the interpretability and qualitative evaluation of the composition discovered by DISC, by manually inspecting the composition on two datasets. In supplementary material, we discuss the interpretability of the composition of ArXiv-abstracts.

First, we consider the *Mammals* dataset provided by the European Mammal Society. This dataset consists of presence records of 124 European mammals within areas of 50-by-50 kilometres. Additional geographical information were not used during the experiments.

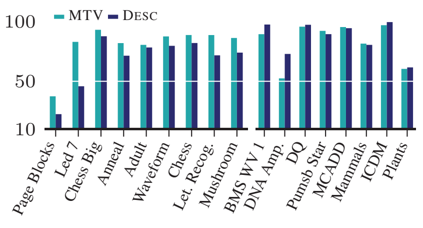
DISC discovers 9 components with 64 patterns in total. We geographically depict the components DISC discovers in Fig. 3. Although it did not know the spatial locations of the data points, it discovers (almost complete) contiguous areas



(a) Number of Patterns

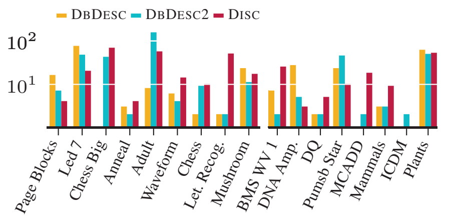


(b) Runtime [s]

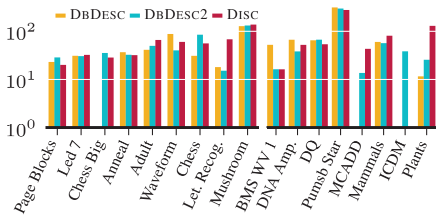


(c) Likelihood Ratio [%]

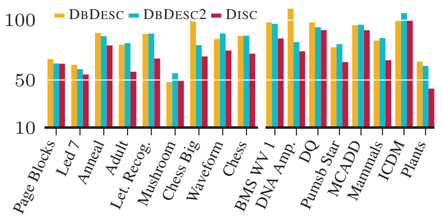
Figure 1: **DESC Efficiently Discovers Concise Pattern Sets.** From left to right, in Fig. 1a the number of discovered patterns (log-scale), in Fig. 1b the runtime (seconds, log-scale), and in Fig. 1c the likelihood ratio $\ell/\ell_\emptyset$ (lower is better). Each figure consists of two groups for labeled (first group) and unlabeled (second group) data.



(a) Number of Components



(b) PSKL



(c) Likelihood Ratio [%]

Figure 2: **DISC Discovers Informative and Interpretable Compositions.** The number of components and the in Fig. 2a the number of discovered components (log-scale), in Fig. 2b PSKL (higher is better), and in Fig. 2c the likelihood ratio $\ell/\ell_\emptyset$ (lower is better) of MTV and DESC. Each figure consists of two groups for labeled (first group) and unlabeled (second group) data.

in Europe that correspond to ground truth habitats. Moreover, the patterns it discovers for these components are meaningful: for example, although the combination of species as *Wolverine* and *Norway Lemming* are highly characteristic for both "Scandinavian" components, their distribution differs between these components. For the Iberian Peninsula the *Common Genet* and *Mediterranean Pine Vole* are discovered to be very characteristic. The habitation zone of the latter spreads to southern France, and this is reflected by this pattern being shared between these two components. Further, DISC discovers that the co-occurrence of *Eurasian beaver, Red squirrel* are descriptive across Europe. Last, but not least, DISC finds that the *Eurasian Harvest Mouse, European Mole, Eurasian Water and Pygmy Shrew, Stoat, Field Vole* are all very common across Europe, and include them in a single pattern shared amongst most components stretching Europe.

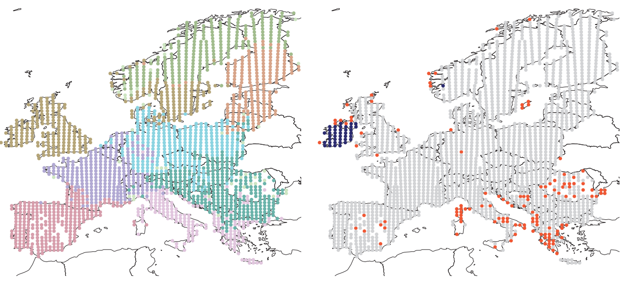


Figure 3: **Discovers Meaningful Partitions.** Results of DISC **(left)** on the *Mammals* dataset. The 9 components represent contiguous areas that correspond to known habitats. DBSCAN **(right)** essentially only discovers Ireland

Discussion

DISC discovers meaningful, easily interpretable composition. The components are described concisely, by characteristic and shared patterns. DISC explains *why* there are components, what makes them special and what is common among them. Our manual inspection show that the results are easy to interpret and informative. In addition, DESC is a highly efficient pattern miner for single and multiple datasets and beats state-of-the-art in descriptiveness, conciseness and runtime. Furthermore we showed how DESC can be used to describe the result of a clustering algorithm. Furthermore, we observe that jointly optimizing for interpretability and likelihood is doable in practice and can outperform clustering algorithms like DBSCAN with a post-hoc explanation.

We studied the novel problem of discovering the composition, that is a partitioning of the dataset and its description using locally characteristic patterns, or patterns shared across sets of components. We formalized this problem in terms of the family of maximum entropy distributions over itemsets and defined the best composition as the one that gives the most succinct description of the data. We introduced an efficient pattern miner DESC for succinctly describing a single or multiple data components. Using this, we described an algorithm to discover the partitioning of a dataset. Both together discover the composition. Experimental evaluation showed that DISC efficiently discovers interesting and meaningful composition of the data.

References

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules. In *VLDB*, 487–499.

Bonferroni, C. E. 1936. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze* 8:3–62.

Budhathoki, K., and Vreeken, J. 2015. The difference and the norm – characterising similarities and differences between databases. In *ECML PKDD*. Springer.

Chakrabarti, D.; Papadimitriou, S.; Modha, D. S.; and Faloutsos, C. 2004. Fully automatic cross-associations. In *KDD*, 79–88.

Chen, J.; Chang, Y.; Hobbs, B.; Castaldi, P.; Cho, M.; Silverman, E.; and Dy, J. 2016. Interpretable Clustering via Discriminative Rectangle Mixture Model. In *ICDM*, 823–828.

Csiszár, I. 1975. I-divergence geometry of probability distributions and minimization problems. *Annals Prob.* 3(1):146–158.

Darroch, J., and Ratcliff, D. 1972. Generalized iterative scaling for log-linear models. *Annals Math. Stat.* 43(5):1470–1480.

Dempster, A.; Laird, N.; and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B* 39(1):1–38.

Dhillon, I. S.; Mallela, S.; and Modha, D. S. 2003. Information-theoretic co-clustering. In *KDD*, 89–98.

Ester, M.; Kriegel, H.; S, J.; and Xu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. 226–231. AAAI Press.

Fowkes, J., and Sutton, C. 2016. A Bayesian Network Model for Interesting Itemsets. In *ECML PKDD*, 410–425. Springer.

Jaynes, E. 1982. On the rationale of maximum-entropy methods. *Proc. IEEE* 70(9):939–952.

Kim, B.; Shah, J. A.; and Doshi-Velez, F. 2015. Mind the Gap: A Generative Approach to Interpretable Feature Selection and Extraction. In *NIPS*. Curran Associates, Inc. 2260–2268.

Lakkaraju, H.; Bach, S. H.; and Leskovec, J. 2016. Interpretable Decision Sets: A Joint Framework for Description and Prediction. In *KDD*, 1675–1684.

MacQueen, J. 1967. Some methods for classification and analysis of multivariate observations. In *Berkeley Symp. Math. Stat. Prob.* Vol. I: Statistics, pp. 281–297.

Mampaey, M.; Vreeken, J.; and Tatti, N. 2012. Summarizing data succinctly with the most informative itemsets. *ACM TKDD* 6:1–44.

Menéndez, M. L.; Pardo, J. A.; Pardo, L.; and Pardo, M. C. 1997. The Jensen-Shannon divergence. *J Franklin Inst* 334(2):307–318.

Nijssen, S.; Guns, T.; and De Raedt, L. 2009. Correlated itemset mining in ROC space: a constraint programming approach. In *KDD*, 647–656. Springer.

Schwarz, G. 1978. Estimating the dimension of a model. *Annals Stat.* 6(2):461–464.

Smets, K., and Vreeken, J. 2012. SLIM: Directly mining descriptive patterns. In *SDM*, 236–247. SIAM.

Tatti, N. 2006. Computational complexity of queries based on itemsets. *Inf. Process. Lett.* 98(5):183–187.

Webb, G., and Vreeken, J. 2014. Efficient discovery of the most interesting associations. *ACM TKDD* 8(3):1–31.

Webb, G. I. 2010. Self-sufficient itemsets: An approach to screening potentially interesting associations between items. *ACM TKDD* 4(1):1–20.