ExClus: Explainable Clustering on Low-dimensional Data Representations

Xander Vankwikelberge, Bo Kang, Edith Heiter, and Jefrey Lijffijt
Ghent University, Ghent, Belgium

Abstract. Dimensionality reduction and clustering techniques are frequently used to analyze complex data sets, but their results are often not easy to interpret. We consider how to support users in interpreting apparent cluster structure on scatter plots where the axes are not directly interpretable, such as when the data is projected onto a two-dimensional space using a dimensionality-reduction method. Specifically, we propose a new method to compute an interpretable clustering automatically, where the explanation is in the original high-dimensional space and the clustering is coherent in the low-dimensional projection. It provides a tunable balance between the complexity and the amount of information provided, through the use of information theory. We study the computational complexity of this problem and introduce restrictions on the search space of solutions to arrive at an efficient, tunable, greedy optimization algorithm. This algorithm is furthermore implemented in an interactive tool called ExClus. Experiments on several data sets highlight that ExClus can provide informative and easy-to-understand patterns, and they expose where the algorithm is efficient and where there is room for improvement considering tunability and scalability.

Keywords: Dimensionality reduction · clustering · explainable AI · exploratory data analysis · hierarchical clustering · t-SNE.

1 Introduction

Artificial intelligence methods exceed human performance on many tasks and thus have found widespread use, yet the resulting models are often black boxes. There is a growing demand to create scalable human-friendly implementations. The creation of explainable white-box artificial intelligence methods is necessary to have users trust, manage, and use these in making crucial decisions [1,8].

This need is also present in the area of clustering and dimensionality reduction methods for high-dimensional data. Clustering and dimensionality reduction methods are frequently employed to get a grasp of the high-level structure

* The research leading to these results has received funding from the ERC under the EU’s Seventh Framework Programme (FP7/2007-2013) (ERC Grant Agreement no. 615517) and under the EU’s Horizon 2020 research and innovation programme (ERC Grant Agreement no. 963924), from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” programme, and from the FWO (project no. G091017N, G0F9816N, 3G042220)
of data. Especially non-linear dimensionality reduction methods such as Isomap [16], LLE [12], t-SNE [10], and UMAP [11] manage to effectively map data onto a low-dimensional space, retaining the relative distances from the high-dimensional space, but this low-dimensional space is often difficult to understand [14].

**Related work.** Possible solutions in several directions have been studied: we may explore the 2D projection by letting the data glyphs correspond to specific attributes (e.g., color the points using the values for a specific attribute) or draw attribute isolines (as in DimReader [7]). Such solutions are limited to a single or very few attributes at a time and it is not obvious how to make them practical for data whose original dimensionality is large (e.g., from 20 attributes or more).

There also exist approaches that focus on data points rather than attributes: ‘Forward projections’ show how points would move in the low-dimensional embedding if their attributes would change, and we may also do the inverse exercise of ‘backward projection’, i.e., how the attributes would need to change to move a point in a certain direction (which does not have a unique solution, so an inductive bias is necessary to resolve this) [3]. However, it is not practical to learn the structure of a large data set by exploring each point individually. ‘Probing clusters’ [15] means to explore the feature values for manually selected sets of points that for example appear to form a cluster. This may lead to insights on the high-level structure. Yet, for high-dimensional data we are still left with the problem that this does not scale to a large number of attributes.

Most similar to our work is the Clustrophile 2 tool [4], which lets users explore a diverse set of pre-generated clusterings by means of a scatter plot showing a low-dimensional projection, as well as an attribute similarity matrix and/or a decision tree for the cluster assignment. However, the decision tree does not explicitly show how each cluster stands out, while the similarity matrix is only a visual aid that puts the burden of the comparison fully on the user, who are likely quickly overwhelmed for data with a large number of attributes.

**Contributions.** We propose in this paper to take a very different approach: to compute an interpretable clustering that facilitates the analysis of a scatter plot of dimensionality-reduced data. We do this in several steps: (1) first we use information theory to quantify the informativeness of the mean and variance statistics for a given subset of points and given subset of attributes. (2) To control the trade-off between the complexity of the clustering and the amount of information gained, we introduce a simple notion of complexity for such a ‘bicluster pattern’, which gives us control of how complicated the explanations for the clusters may be. (3) Integrating this with an approach to cluster the data on the low-dimensional representation, we obtain coherent clusters on the dimensionality-reduced scatter plot along with a subset of attributes that are informative (in the information-theoretic sense), to explain this cluster. (4) Using this, we attempt to automatically generate an optimal explainable clustering on low-dimensional representations of data sets, so that users can meaningfully explore complex data with a limited time cost for interpreting apparent structure.

The contributions of this paper are as follows:
We define bicluster patterns, their informativeness using information theory and a simple notion of their interpretational complexity.

We derive an algorithm to optimally cluster a dimensionality-reduced scatter plot such that the clusters are as informative as possible using a few attributes in the high-dimensional space.

We study the computational complexity of this problem and argue for the use of hierarchical clustering to reduce the search space of possible solutions.

We provide an implementation of that algorithm as well as a browser-based tool called ExClus (see Fig. 1) that can be used to explore data in practice.

We present experimental results on a few datasets to explore the usefulness of the tool as well as the effect of the two hyperparameters that govern the interpretational complexity of bicluster patterns. We present some early feedback from users and empirically study the scalability of the method.

We find that ExClus has great potential to further facilitate data exploration with dimensionality-reduction methods and that the method is sufficiently scalable to use also on large data.

The ExClus tool is freely available at https://github.com/aida-ugent/ExClus

Fig. 1: The ExClus user application. Results are generated by applying the algorithm on the UCI Adult data set, an extract from the 1994 USA census [6].
The paper is structured as follows: in Section 2 we first formalize the type of patterns that we aim to extract and define the informativeness and descriptional complexity of these patterns. We then introduce the optimization algorithm to extract the patterns. In Section 4 we introduce ExClus and the user interface. Experiments are given in Section 5 and conclusions are presented in Section 6.

2 Method

The general idea is to define the substructure that we aim to find as a pattern, which we call a bicluster pattern. Informally, this is a subset of points and the mean and variance statistics for a subset of the attributes. The approach then builds upon the framework for subjective interestingness for patterns [5], which suggests the use of information theory to quantify how much information a user gets per time unit spent. There are three key concepts in this framework: information content, description complexity, and subjective interestingness.

The information content expresses how much the user learns by showing them a specific pattern. The description complexity aims to express the difficulty of understanding the pattern, and the subjective interestingness is simply the ratio of the two. The term ‘subjective’ explicates that how much we learn can only be specified with respect to prior expectations over the data. In practice we will use a prior based on the data, but it may also be chosen subjectively. Barring issues of feasibility, the prior could reflect the actual knowledge of a user.

We discuss each of these concepts in more detail: In Section 2.1, we express the information content of a bicluster pattern, i.e., the number of bits of information that we learn by showing the statistics for this bicluster, in comparison to given prior expectations. In Section 2.2, we also introduce the description complexity that aims to capture how difficult or time consuming it is to process the presented information, for a human end-user. In Section 2.3, we then formalize the explainable clustering problem: to find a set of bicluster patterns that partition the data into clusters that are coherent with respect to a given 2D projection, such that the subjective interestingness of the clustering is maximized.

2.1 Bicluster patterns and their information content

Notation. Let \( \mathbb{X} \) be an \( n \times m \) data matrix with \( X_i \) denoting data point \( i \in \{1, \ldots, n\} \) and \( X_{ij} \) the value for the \( j \)-th attribute \( (j \in \{1, \ldots, m\}) \) of \( X_i \). We write \( t(j) \in \{\text{bool}, \text{real}\} \) to denote the type of attribute \( j \).

Definition 1. A bicluster pattern \( \mathcal{P} \) is a tuple \((D, A, S)\) with \( D \subseteq \{1, \ldots, n\} \) a set of data points indices, \( A \subseteq \{1, \ldots, m\} \) a set of attribute indices, and the statistics \( S = \{S_{A_1}, \ldots, S_{A_{|A|}}\} \), corresponding to the attributes whose indices are in \( A \). For boolean attributes \((t(j) = \text{bool})\), \( S_j \in [0,1] \) is a frequency and for real-valued attributes \((t(j) = \text{real})\) both a mean and standard deviation: \( S_j \in \mathbb{R} \times \mathbb{R}^+ \).

To express how much we learn by observing the statistics for a set of attributes for a subset of the data, we first need to express what we are comparing
against, i.e., we have to define a prior distribution for the data. We take a simple approach here and use as prior the maximum likelihood statistics fitted on the full data, without accounting for any co-variate structure. That is, each attribute is assumed to be independent. Although in some context it may be preferable to already account for co-variates between the attributes, in many cases the independence assumption is good as it is also transparent for the user.

Similarly, the information that we obtain by observing a bicluster pattern are the maximum likelihood statistics $S$ for the set of points $D$ and the attributes $A$. The statistics may be used to derive a Maximum Entropy model for the data. As the statistics are assumed independent, indeed also the Maximum Entropy model is independent and we may write it for each attribute separately. It depends on the attribute type, so for brevity we write this model as $M_{t(j)}(S_j)$. The Maximum Entropy model is a Gaussian distribution for real-valued attributes and a Bernoulli distribution for boolean attributes.

The information content of a pattern $P$ is equivalent to the Kullback-Leibler divergence between the prior expectations and the statistics contained in $P$:

$$I(P) = \sum_{i=1}^{\mid D \mid} \sum_{j=1}^{\mid A \mid} D_{KL}(M_{t(A_j)}(S_{D_{A_j}})||M_{t(A_j)}(S_{X_{A_j}})). \tag{1}$$

Note that we overloaded $D_{KL}$ here to refer to the KL divergence, but in all other occurrences $D$ indeed refers to a subset of the data points included in a bicluster pattern. The KL divergences are straightforward to compute analytically for both the Bernoulli and Gaussian distribution.

Note that it may happen that the variance of a bicluster for a specific real-valued attribute is zero, in which case the KL divergence will be infinite. Therefore, we add a small value $\epsilon$ to all variance estimates. Resolving this in a more robust manner by for example considering the precision of the real-valued numbers is left for future work.

### 2.2 Description complexity

The aim of the description complexity is to quantify how difficult it is to process the presented information, i.e., how time consuming it is to internalize. Unfortunately we have no realistic models of human cognition so we will just need to work on assumptions. It is important to realize that our aim here is not to do model selection in the statistical sense and we are not presenting the patterns to another computer. The aim is simply to have a formula that is suitably parameterized such that we can balance the amount of information and the complexity of the identified patterns.

In previous papers on subjective interestingness, the description complexity was quantified as a linear function over the number of statistics that is presented to the user. Often, this leads to a more tractable optimization problem. We instead choose to make it explicit that providing more statistics realistically has a superlinear effect on the amount of time to process the information. Because
we are going to present the user not a single bicluster pattern, but a set that partitions the data, we define the description complexity directly for a set of bicluster patterns as $\alpha$ plus the total number of statistics to the power $\beta$:

$$ C(\{P_1, \ldots, P_k\}) = \alpha + \left( \sum_{i=1}^{k} \sum_{j=1}^{|A^{P_i}|} |S_{A_j}^{P_i}| \right)^\beta. \quad (2) $$

Here $|S_{A_j}| = 1$ for boolean attributes and $|S_{A_j}| = 2$ for real-valued attributes. This formula also includes two hyperparameters, $\alpha$, and $\beta$, which allow for tuning by users and enable the identification of more intuitive solutions.

### 2.3 Explainable clustering problem

Finally, we want to find a clustering that makes a trade-off between the total information content and description complexity. Hence, we define the subjective interestingness for a set of bicluster patterns as

$$ S(\{P_1, \ldots, P_k\}) = \frac{\sum_{i=1}^{k} I(P_i)}{C(\{P_1, \ldots, P_k\})}. \quad (3) $$

The missing component is that we have not related these concepts yet to the low-dimensional projection that we started out from. Let $Y$ denote this low-dimensional—typically 2D—projection of $X$. We may now define the explainable clustering problem.

**Problem 1.** The explainable clustering problem is to find a set of bicluster patterns $\{P_1, \ldots, P_k\}$ ($k \geq 1$) that partition the data (i.e., they cover all data points $\bigcup_{i=1}^{k} D_i = \{1, \ldots, n\}$ and no data point is covered twice $D_i \cap D_j = \emptyset \forall i \neq j$) in a coherent manner with respect to $Y$ and that maximize the subjective interestingness $S(\{P_1, \ldots, P_k\})$.

Here we have not defined exactly when a partitioning may be called coherent with respect to $Y$. Indeed, this may differ per usage scenario and we argue that it is not obvious there may exist a universally applicable definition. We will argue for a particular choice in the following section that also benefits the design of an efficient optimization algorithm.

### 3 Search algorithm

We are faced with a difficult optimization problem: we want to cluster the data in a given low-dimensional projection, in such a manner that the attribute values are coherent and we are concurrently selecting attributes to explain the clusters.

We first considered a two-step approach; first clustering the data and then computing the optimal explanations. Alternatively, we could derive an iterative optimization scheme, e.g., an EM scheme switching between data point assignment and the explanation attributes. However, we opt for a third solution where we constrain the search space to make it feasible to optimize the cluster assignment, number of clusters, and the attribute selection in an integrated manner.
3.1 The ExClus algorithm

Step 1. We start by computing a hierarchical agglomerative clustering using the Euclidean distance on the projection $\mathsf{Y}$. A tested and popular method for clustering data, this conveniently ensures coherence of the final clustering solution and it constrains the search space. The resulting dendrogram will be used to guide the full clustering, but we will not use the distances in $\mathsf{Y}$ to select clusters.

Step 2. Given the optimization problem, we have to consider how to cut the dendrogram to obtain the best set of bicluster patterns. We are still faced with a large number of possible clusterings. If the tree is sufficiently large and balanced, the number of possible clusterings with $k$ clusters is given the corresponding Catalan number, which is already 1430 for $k = 8$, for example.

To obtain an approximate solution, we optimize the clustering by splitting one cluster at a time. That is, we start with everything in one cluster, then consider all possible splits for $k = 2$. This can be done in time linear in the size of the dendrogram, as indeed we may exactly split any branch off. For each possible split, we can optimize the attributes by ranking them and greedily considering whether we should add a boolean or a real-valued attribute (because they weigh differently in the description complexity). We obtain indeed the optimal solution for $k = 2$ given the constraints, but then we take the $k = 2$ solution and split this once more in any position in the tree to obtain a solution for $k = 3$, which is not guaranteed to be optimal. The procedure continues for $k > 3$.

This procedure could go on until the number of clusters is equal to the number of samples. This would be time consuming and it is highly unlikely that a solution with very large $k$ would maximize the subjective interestingness. Hence, we implement a time limit to ensure the runtime stays in an acceptable range. This limit can be any value, from a few milliseconds to only stopping when all calculations end.

3.2 Solution refinement

It is not obvious how to tune the hyperparameters, so we expect this to be done through trial-and-error. From experiments presented in Section 5, we find they can have a dramatic effect on the resulting solution and the differences may also be abrupt. Restarting the algorithm from scratch is then not always desirable to find a good solution. Hence, we also introduce a procedure to modify the current solution given new hyperparameter values.

This modification again uses the dendrogram, but instead of starting at one cluster, it starts from the previously obtained result. The procedure then evaluates both whether splitting more clusters off, as described in Section 3.1, or merging some clusters back together that were previously split off results in a higher subjective interestingness.

4 User application

In this section we present the user interface of ExClus, which is based on the previously introduced formalization and algorithm. The application allows test-
ing and intuitive usage of the presented algorithm. In the following, we highlight different aspects of the interface and summarize user feedback.

4.1 Design choices

The interactive tool helps users to understand data sets by generating clusters and their explanations simultaneously. Users can browse through the explanations at their own pace and tune the algorithm to retrieve more insights. The application is created with Dash, a Python framework for building data science web applications. Figure 1 shows the user interface.

**Dashboard.** The dashboard, situated at the top of the application, shows some essential information on the currently presented clustering. After tuning the hyperparameters to get different results, it also shows how these values changed. Alone, these values do not tell much, but after tuning, it is helpful to compare how they changed.

- **# clusters:** How many clusters the algorithm generated.
- **# attributes:** Each cluster needs a certain number of attributes as an explanation (at least one). For example, cluster one might need only one attribute, but cluster two might need four. This number is the sum of all these attributes for all clusters.
- **Information content:** This is the entire clustering’s Information Content. It is a deliberate choice to show the Information Content and not the Subjective Interestingness score, as the latter depends on the values of the hyperparameters. It is not sensible to compare those values across different hyperparameter choices, while the Information Content can indeed be compared.

**Data Embedding.** In the top left of the interface we show a scatter plot of the low-dimensional data and display the clustering result from ExClus with different colors. Going from multiple dimensions to only two with a non-linear transformation, which dimensionality reduction techniques as t-SNE do, results in difficult to interpret axes, so the only thing deductible from the graph, besides the clustering, is that points close together on the 2D plot are similar in the high-dimensional space.

**Cluster explanations.** This part of the application provides the identified explanations of why specific points form a cluster. When users select one cluster using the drop-down menu, we show the relative size of the cluster and the description of attributes that make this cluster distinct from the rest of the data. We sort the attributes by decreasing information content and use different visualizations for binary and real-valued attributes as shown in Figure 2.

For binary attributes, we illustrate the distribution of values in a stacked bar chart. The left bar shows the distribution within the selected cluster and the right bar the distribution of the entire data set. For real-valued attributes, we derived the information content by fitting Gaussian distributions using mean and
standard deviation as parameters. To compare the selected cluster to the entire dataset, we plot both probability density functions for the Gaussian distributions in the prior model and the model of the cluster.

**Hyperparameter tuning.** Below the data visualization we display the current parameter values that influence the number of clusters and the detailedness of their explanations. Range sliders allow the user to change $\alpha$, $\beta$, and the runtime limit of the greedy search. A more extensive look into the effects of these parameters follows in Section 5. Next to the parameter sliders we provide two ways of applying the new parameter values: *refine* and *recalc*. The refine option starts the algorithm from the current clustering as described in Section 3.2. The recalc option starts the algorithm from scratch. That way, users can either build on the current clustering and understanding of the data or investigate a new approach.

5 Experiments

In this section we discuss an empirical evaluation that we conducted to test ExClus. First we present the results from case studies on three data sets, secondly we consider quantitatively the effects of the hyperparameters, and finally we discuss experiments on the scalability of the method.

5.1 Use cases

This section gives a short evaluation of the algorithm’s results to give an initial insight into the application’s possibilities.

**UCI Adult.** The UCI Adult data set is an extraction from a 1994 USA census database, and the attributes included are: age (continuous), gender (male/female), ethnicity (white/other), education level (continuous), hours worked per week (continuous), and income ($\leq 50k$, $> 50k$). This experiment uses a sample of only 2500 of the nearly 32000 data points for performance reasons. An example of the algorithm’s result is visible in Figure 1.
Dissecting the clustering reveals that the dimensionality reduction method mostly used three attributes to create the clusters: gender, income, and ethnicity. However, by tuning the hyperparameters, the algorithm can split these clusters further to reveal more details on the people included in each group or put clusters back together to show they are part of a more general pattern. For example, cluster two (green) contains all the white males with an income above $50k, which is more than 30% of the data set. There is a possibility of splitting this cluster further to distinguish between attributes such as education level or hours worked per week by reducing $\beta$. However, this can overwhelm a user initially, so it is better to start with few clusters and then refine them to learn more details.

Other significant sections in this clustering are cluster zero (blue), which are the female counterparts of cluster two, and this cluster’s explanation is also partially shown in Figure 1. The top right then contains all the high-income white people, with the red cluster being females and the other two (black and magenta) males, where the algorithm further separates them on high education level (black) and average education level (magenta). This difference also reveals that the dimensionality reduction does not reveal every pattern because there is no such split for the females as there are not enough high-income females included for the dimensionality reduction method to emphasize it. Finally, the bottom right has three clusters, including only other than white ethnicities, and the algorithm split them in almost the same manner as the white people, but again with less detail as there are not that many of them in the data sample.

**German socio-economic data.** This data set includes the socio-economic information of 412 German districts [2]. There are more than thirty attributes, but many of them are related, and overall, there are three main categories. First, the voting record attributes contain the voting percentages for the five largest parties (Green, Left, CDU, SPD, and FDP) in the 2005 and 2009 elections, where we included only the latter attributes. Another block of attributes defines the age distribution for the district, such as the percentage of old or young people. The remaining attributes give information on the workforce, such as which percentage of people work in which sector (agriculture, finance, service, etc.).

The application’s outcome for two different parameter settings, shown in Figure 3, highlight a challenge in making ExClus effective. Notably, for hyperparameter settings that gave solutions with only a few clusters on other data, the solution here has two clusters with only a tiny number of data points (cluster zero and one). On the UCI Adult data this happened only with extreme parameter settings. Changing the hyperparameters here worsens this issue. On the one hand, a user could solve this by changing the hyperparameters of the dimensionality reduction method, t-SNE in this case, but it can be a strenuous task to test different embeddings on the application each time. Furthermore, assuming that the low-dimensional representation is given, we would prefer that the application deals with this problem regardless of the embedding. It appears the problem is at least partly related to the dendrogram of the agglomerative hierarchical cluster. Specifically, the problem disappears if we consider another linkage criterion, where we used single linkage in other cases, here complete or
average appears to be preferable. The single linkage appears to fail here because the clusters are not as homogeneous as for other data sets.

Interestingly, looking at the clusters in Figure 3b and highlighting the districts on a German map, they align almost entirely with the patterns discovered in previous research by Lijffijt et al. in their research on finding subjectively interesting subgroups in data sets with real-valued attributes [9]. Comparing some of the attributes included in the explanations of the clusters further proves that these patterns are almost equal. Take, for example, cluster two (green), which maps to pattern b in Figure 4. Just as in the paper, this cluster consists of districts with a higher population density, indicating they are cities, with a political favor towards the Green party and a large percentage of middle-aged people.

**Cytometry data.** Cytometry measures the characteristics of cells and has a broad range of applications. In this case, the experiment uses single-cell data of one mouse also used in previous research, which looked at different ways to make sense of increasingly higher dimensional data retrieved with a specific cytometry technique [13]. Each cell is described by nine markers and the researchers in
Figure 5 (left) shows the results directly obtained from the original research paper. This dimensionality reduced version of the data set, calculated with t-SNE, has colored labels for each data point. The markers indicate which type of cell it is. These results are similar to the ones obtained from ExClus (Figure 5, right). The two large clusters are immediately distinguishable as similar and the smaller clusters also align almost perfectly. For example, the blue cluster from the ExClus results corresponds to the purple cluster from the original paper and the magenta cluster corresponds to the blue cluster.

Furthermore, ExClus simplifies the explanations. In the original cytometry paper color maps, showing the expressions of each gene marker, allows for analyzing the dimensionality reduced clustering. On the other hand, ExClus selects the most important attributes and represents them as a distributions compared to the data set’s average. For example, to evaluate the blue cluster using the original method a user would have to examine all nine color maps while the ExClus algorithm only presents two attributes.

This case study shows that ExClus can explain the data set to the same extent as the paper, but with a lower descriptive complexity as a user does not need to scan all the different marker figures, which eventually simplifies and speeds up the process of understanding a data set.

5.2 Hyperparameters

Both parameters affect the number of clusters and detailedness of explanations, but they do this differently. $\alpha$ is a startup cost for the description length allowing for a more detailed explanation and more clusters as it results in the need for a larger description length before having a substantial impact on the ratio. This parameter’s value has values between zero and a thousand. On the other hand, $\beta$ serves as a penalty for the description length and usually has a value between...
one and two. If the value is larger than one, this applies a penalty that increases super-linearly with an increasing description length, forcing a faster cutoff on explanation and clustering.

Figure 6a shows the effect of $\alpha$ and $\beta$ on the number of clusters and attributes. This graph reveals an almost linear increase in the number of clusters and attributes, where $\beta$ divides the values into different sections, and $\alpha$ ensures a further evolution within each section. Furthermore, Figure 6b is a testament to why these parameters are necessary. It presents the number of clusters (y-axis) in the clustering the ExClus algorithm decided on for different values of $\alpha$ (x-axis) and $\beta$ (color). If $\alpha$ is zero, there are almost no clusters, and changing $\beta$ does not affect the results. Besides that, if $\beta$ is one (no penalty), $\alpha$ changes the results almost uncontrollably (blue graph). While $\alpha$ and $\beta$ might not have a straightforward relationship and interaction, they are both necessary and allow users to change the number of clusters and the number of attributes.

### 5.3 Scalability

Data sets can scale in two dimensions. On the one hand, the number of data points can increase, and on the other hand, the number of features can increase. Both can have severe effects on the runtime, and they should be taken into consideration whenever the algorithm runs. The algorithm goes through multiple iterations of the greedy optimization step. Each iteration searches for the optimal clustering with the number of clusters equal to the iteration. When the runtime ends, it selects the iteration’s outcome with the highest subjective interestingness. Therefore, it is best to define scalability in the number of iterations the algorithm reaches. Figure 7a presents these effects when the number of data points increases and Figure 7b when the number of features increases.
(a) Number of iterations vs. number of data points on the Cytometry data. 
(b) Number of iterations vs. number of features on the Gene expression data.

Fig. 7: Number of iterations the algorithm reaches within specified runtimes.

While both figures show an exponential decrease in iterations, this only becomes a noticeable issue if the data set is enormous (more than 100,000 data points or more than 500 features). Furthermore, in most cases, some sampling or feature importance selection will occur if the data set is this large because dimensionality reduction methods and hierarchical clustering techniques can also suffer from scalability issues. Besides that, it is unlikely for these data sets that a clustering with a vast number of clusters will be the optimal result as understanding more than a hundred patterns is highly complex. However, while it is not a limiting issue in most cases, the algorithm does have room for improvement in certain areas that could increase performance.

6 Conclusion

This paper introduced a new method to assist users in analyzing high-dimensional data sets. It contributes to the state of the art in considering how to find a good balance between informativeness and the complexity of information presented to the user, by making a trade-off between information and its descriptive complexity. Specifically, while generating clusterings and their accompanied optimal explanations automatically. The presented algorithm to identify an explainable clustering on top of a scatter plot of dimensionality-reduced data is implemented in a publicly-available tool called ExClus.

From case studies we have observed that ExClus can be used to effectively analyse data, by comparing results with previous studies on that data (German SE and Cytometry data), highlighting ExClus enables identification of previously known and possibly new patterns with little effort.

Further study could include investigation in methods to choose or support users in choosing good values for the hyperparameters. ExClus allows quick
experimentation with hyperparameters, but for now it remains an exercise of trial-and-error for users of the system. Secondly, we have omitted from the scope of this study which dimensionality-reduction method to use. We have used t-SNE in all experiments, which is not the most scalable algorithm and about which many critiques have been written. It would be worthwhile to investigate which dimensionality-reduction methods would synergize best with ExClus. Finally, we have observed that the linkage criterion can be important to consider. More generally, it may be interesting to study other ways to build a restricted search space like the agglomerative clustering approach used here.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). IEEE Access 6, 52138–52160 (2018)
2. Boley, M., Mampaey, M., Kang, B., Tokmakov, P., Wrobel, S.: One click mining: Interactive local pattern discovery through implicit preference and performance learning. In: Proc. of KDD-IDEA Workshop. pp. 27–35 (2013)
3. Cavallo, M., Demiralp, Ç.: A visual interaction framework for dimensionality reduction based data exploration. In: Proc. of CHI (2018)
4. Cavallo, M., Demiralp, Ç.: Clustrophile 2: Guided visual clustering analysis. IEEE TVCG 25(1), 267–276 (2019)
5. De Bie, T.: Subjective interestingness in exploratory data mining. In: Proc. of IDA. pp. 19–31 (2013)
6. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
7. Faust, R., Glickenstein, D., Scheidegger, C.: DimReader: Axis lines that explain non-linear projections. IEEE TVCG 25(1), 481–490 (2019)
8. Gunning, D., Aha, D.W.: DARPA’s explainable artificial intelligence (XAI) program. AI Magazine 40(2), 44–58 (2019)
9. Lijffijt, J., Kang, B., Duivesteijn, W., Puolamäki, K., Oikarinen, E., De Bie, T.: Subjectively interesting subgroup discovery on real-valued targets. In: Proc. of ICDE. pp. 1352–1355 (2018)
10. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. JMLR 9, 2579–2605 (2008)
11. McInnes, L., Healy, J.: UMAP: Uniform manifold approximation and projection for dimension reduction. ArXiv:1802.03426 (2018)
12. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
13. Saeyts, Y., Van Gassen, S., Lambrechts, B.N.: Computational flow cytometry: helping to make sense of high-dimensional immunology data. Nature Reviews Immunology 16(7), 449–462 (2016)
14. Sedlmair, M., Brehmer, M., Ingram, S., Munzner, T.: Dimensionality reduction in the wild: Gaps and guidance. Tech. Rep. TR-2012-03, University of British Colombia, Vancouver, BC, Canada (2012)
15. Stahnke, J., Dörk, M., Müller, B., Thom, A.: Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions. IEEE TVCG 22(1), 629–638 (2016)
16. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)