EduClust – A Visualization Application for Teaching Clustering Algorithms

J. Fuchs[1] †, P. Isenberg[2], A. Bezerianos[2,3], M. Miller[1] and D. Keim[1]

[1]University of Konstanz, Germany [2]Inria, France [3] Université Paris-Sud & CNRS, France

Abstract

We present EduClust*, a visualization application for teaching clustering algorithms.* EduClust *is an online application that combines visualizations, interactions, and animations to facilitate the understanding and teaching of clustering steps, parameters, and procedures. Traditional classroom settings aim for cognitive processes like remembering and understanding. We designed* EduClust *for expanded educational objectives like applying and evaluating. Educators can use the tool in class to show the effect of different clustering parameters on various datasets while animating through each algorithm's steps, but also use the tool to prepare traditional teaching material quickly by exporting animations and images. Students, on the other hand, benefit from the ability to compare and contrast the influence of clustering parameters on different datasets, while seeing technical details such as pseudocode and step-by-step explanations.*

CCS Concepts

• ***Theory of computation*** → ***Unsupervised learning and clustering;*** • ***Applied computing*** → ***Interactive learning environments;***

1. Introduction

Models for teaching and learning in higher education are heavily debated and frequently changing. Current approaches for teaching are changing from transmitting previously prepared information to dynamic teaching and learning sessions. Dynamic classrooms are characterized by a close interplay between students and teachers and frequently involve a blend of computer and traditional teaching material. Visualizations of abstract concepts, data, or processes can be important facilitators to learning and greatly enhance traditional teaching material in dynamic classrooms. Visualizations for education can come in a range of forms and sophistication: from small visualizations of concepts and phenomena that often accompany text, such as Victor's explorable explanations [Vic18] or Bostock's Explorables [Bos18] to fully interactive teaching and learning tools.

EduClust is an online visualization application we designed to be used as part of dynamic teaching and learning experiences (`http://educlust.dbvis.de/`). *EduClust* supports the teaching of clustering algorithms as part of university courses, such as data mining or visual analytics, by providing lecturers with expanded technological capabilities to accompany or extend their current practices. The application can currently be used to teach or learn 9 common clustering algorithms. It shows the evolution of clustering steps for clusters of 2D data points and highlights accompanying pseudocode. In addition, students and teachers can interactively choose input parameters and datasets and observe and compare the resulting clustering behavior. As an online tool, *EduClust* can be used by students to learn alone with accompanying (traditional) teaching material. It can also be used by lecturers in dynamic teaching sessions, allowing them to discuss with students about possible effects and changes based on various settings for each algorithm. In addition, *EduClust* offers to prepare traditional teaching material from freely chosen (and domain specific) data sets and providing exports of clustering animations to be used in slides and other teaching supports.

We designed *EduClust* considering Krathwohl's [Kra02] taxonomy of educational objectives. We support multiple cognitive processes involved in learning such as *remembering* learned knowledge and *understanding* presented material, as well as higher-level cognitive processes like *analyzing* the structure or relationship of algorithms, *evaluating* parameters and their standards, and *creating* novel data sets and clustering outputs. As such, *EduClust* contributes (a) a visual analytics application that aids both teaching and learning; and (b) an application that represents the processes of multiple complex clustering algorithms, all in one place.

2. Related Work

Our work was inspired by interactive visualizations used in education more generally, as well as specific visualization techniques for clustering algorithms and their use in practice.

† fuchs@dbvis.inf.uni-konstanz.de

DOI: 10.2312/eged.20191023

2.1. The Use of Interactive Visualizations in Education

Several visualization systems have been recently conceived in the context of education, e.g., to improve the navigation of educational concepts and material [SST*17] or for managing massive open online courses [CCZ*16, CCL*16, WYD*16]. In these contexts, visualization systems often support the analysis of learner behaviors to help improve individual courses. A recent example is DropoutSeer, a tool to visually support instructors and machine learning researchers in analyzing the reasons for dropout behavior of students [CCZ*16]. In contrast to this work, we use interactive visualizations as a teaching aid for imparting complex knowledge. Such methods can be as easily used in online settings as in traditional classrooms.

In this context, *EduClust* follows earlier research on algorithm visualization within teaching (summarized by Hundhausen et al. [HDS02] and Shaffer et al. [SCA*10]). Recently, Gomez et al. [GASGTSGP10] highlighted the importance of visual representations in a standardized form within learning processes. Similarly, a past study showed [AH13] how computer science students profit from visualizing concepts, like workflows, within lectures. Especially for the representation of algorithms, studies have shown the benefits of using animation [LBS94], or interaction [HNS00, Sim97] in teaching. Smaller online examples include Victor's explorable explanations [Vic18] or Bostock's Explorables [Bos18] that cover concepts ranging from energy consumption to coding.

The *Seeing Theory* project [Kun16] makes statistics visually accessible. This set of online visualizations explains, for example, how different basic statistics work and what their results are; based on randomly generated or predefined data. In contrast to this work, our approach allows to grasp the stepwise operations of different clustering algorithms, instruction by instruction, including the option to step back and forth. Halim et al. [HKLH12] pursue a similar step-wise approach, but for the explanation of sorting algorithms and other basic computer science topics, like tree traversal strategies [SF11]. We face algorithms of higher complexity: clustering algorithms depend on various starting parameters and typically rely on further data structures that need to be visualized and understood. Finally, in addition to clarifying clustering algorithms for students and allowing self-study after class, our tool helps educators by offering additional ways to prepare material, both before class (images or videos exported from our tool) or on-the-fly in the classroom.

2.2. Visualization Techniques for Clustering Algorithms

Clustering results are commonly visualized as an overlay on existing data visualizations. Overlays often use color or convex hulls to show cluster membership on scatterplots, graphs, or projections of MD points onto the 2D plane. Several such cluster visualizations shown in sequence can demonstrate the evolution of assigned clusters, as previously shown for the K-Means algorithm [Mac67]. Other clustering algorithms rely on additional visual components, like the dendrogram view used in hierarchical clustering [GR69], or the reachability plot for OPTICS [ABKS99]. Next, we discuss common clustering algorithms and their visualization challenges.

2.2.1. Partitioning Methods

K-means is a partition-based clustering algorithm, known for its simplicity and efficiency. The algorithm is initialized with the expected number of clusters in the data. Each cluster is then represented as a prototype (cluster center) positioned at first randomly in data space—how exactly depends on the implementation. The algorithm runs iteratively in two steps. First, the data points are assigned to the closest cluster prototype depending on a distance measure. Second, the prototypes are recomputed, and moved towards a new location taking into account the average distances of the assigned data points. These two steps repeat until convergence.

This repeating process has been visualized using a scatterplot implementation in many projects found online [Moh18, Har14, Sai13]. Data points, as well as cluster prototypes (centers) are represented as single dots in 2D space. In the first step, data points are typically colored based on their cluster membership. In the second step, prototypes are recomputed and repositioned using animation.

Additional features can be added to improve the understanding of this algorithm. Dividing planes [Har14] or lines connecting points to cluster prototypes [Sai13] are design alternatives to better communicate the cluster membership of data points. Independent from implementation, all visualizations we found when researching online tools [Moh18, Har14, Sai13] share two core features: coloring of data points, and moving cluster prototypes using animation to show the repeated process until it converges.

2.2.2. Hierarchical Methods

For agglomerative hierarchical clustering, such as single-, average-, or complete-linkage, the dendrogram is a common representation, and is included in most statistical software solutions like R [R D08], KNIME [BCD*09], WEKA [FHH*09], or jmp [Inc89].

The dendrogram is commonly visualized as a node-link tree showing the distances between clusters, derived from the minimal spanning tree between the data points [GR69]. The branches of the tree represent the clusters, whereas the leaves encode the single data points. The height of the tree is used to connect branches whenever the distance between the respective clusters is met. The tree grows in height until there is only one branch left (the root representing the entire data as one cluster).

Most commonly, dendrograms are represented statically and do not support interactive exploration [BCD*09, FHH*09, R D08, Inc89]. The user has to interpret the plot and then return to the console in order to retrieve the cluster membership of the data points. Yet, recently interactive dendrogram visualizations have also emerged. For example, additional packages enrich the standard R-implementation of the dendrogram with the possibility to cut the tree through direct-manipulation [SHFB17]. Python developers can also profit from a zoomable dendrogram implementation, helping the analyst to focus on specific branches of the tree [Inc]. In *EduClust* , we implemented a similarly interactive dendrogram visualization that works with different data sets and input parameters (e. g., multiple distance metrics and linkage variations).

2.2.3. Density-based Methods

Different types of density-based clustering algorithms exist. Famous representatives are DBSCAN [EKS*96] and OPTICS [ABKS99], which share a similar concept of density. DBSCAN defines density as the number of data points (minPoints) in a given region (epsilon

distance). OPTICS follows a similar approach, introducing the core- and reachability distance to define dense areas. Other clustering algorithms like DENCLUE [HK*98] represent data points as kernel functions, using a more mathematical definition of density. In our work, we focus on DBSCAN (and OPTICS) because of its popularity. We will extend *EduClust* in the near future to further algorithms.

To explain DBSCAN, the authors of the original paper [EKS*96] visualized data points as single 2D dots surrounded by a circle that highlights the epsilon distance. An online implementation of DBSCAN [Har15] uses the same encoding but adds animation and color to communicate the stepwise process and the cluster memberships.

In the original paper, the results of the OPTICS algorithm were displayed in a scatterplot together with the corresponding reachability plot [ABKS99]. The reachability plot helps to better understand regions with varying density within the data by plotting the reachability distance and the core distance of each data point in a bar chart like representation. For further clarification, the corresponding scatterplot can be enriched with additional visual features like the spanning tree of the data points [Wik17]. Breunig implemented a Java applet showing the OPTICS algorithm in action [Bre99]. Like in the original paper, a scatterplot is represented next to the reachability plot. Animation is used to show which data item is currently processed, by highlighting the data point in the scatterplot and its corresponding reachability distance and core distance in the reachability plot. Unfortunately, Java applets are no longer supported by some web browsers, limiting its visibility.

2.3. Searching for Current Practices

In order to derive requirements for the design of our online teaching tool, we began by scanning existing teaching material and interviewing teachers of data mining lectures. In particular, we scanned lecture material from four Universities (Washington: "Cluster Analysis in Data Mining", Illinois: "Machine Learning: Clustering&Retrieval", Stanford: "Machine Learning" and Trento: "Data Mining for Knowledge Management") available on the online learning platform coursera (https://www.coursera.org/, retrieved March 2018) or on the Universities' webpage. We also contacted data mining lecturers to learn about their teaching habits. Four (from major universities in 4 European countries) answered and participated in an interview about their most commonly used visualization techniques.

Table 1 shows a summary of this non-exhaustive review (note that not all clustering algorithms are taught in all classes). Scatterplots were the most prominent choice for visualizing clustering results; independent of the underlying algorithm. In particular, for partition-based clustering, scatterplots were the only visual representation. For density-based methods, scatterplots were again a prominent choice. Only one other representation, the heatmap, was included in the reviewed lecture material. For hierarchical clustering, dendrograms were used by everyone, with scatterplots coming second.

This brief survey helped us limit the visulizations considered for *EduClust*. Since we did not want to focus on new visualization techniques, we constrained our design space to familiar visualizations already used in the teaching of clustering algorithms.

Table 1: *Overview of the visualizations used in online lecture material across different Universities and interviewed lecturers.*

Visualization Technique	K-Means	Hierarchical Clustering	DBSCAN	OPTICS
Scatterplot	8/8 (100%)	5/6 (83.3%)	5/5 (100%)	1/1 (100%)
Dendrogram	0	6/6 (100%)	0	0
Reachability Plot	0	0	0	1/1 (100%)
Heatmap	0	0	1/5 (20%)	0

2.4. Summary

Several visualization prototypes for investigating clustering processes are available online. However, easily comparing different clustering algorithm outputs or the influence of input parameters on various datasets is mostly unsupported. The goal of *EduClust* is to facilitate education about multiple clustering algorithms in one application and to allow for a comparison between different methods, input parameters, and datasets.

3. Educational Objectives & Methodology

EduClust is built upon a theoretic foundation, to follow appropriate educational objectives for learning clustering algorithms. A widely known taxonomy of educational objectives is Bloom's Taxonomy [BEF*56]. It was conceived as a "panorama of the range of educational possibilities against which the limited breadth and depth of any particular educational course or curriculum could be contrasted" [Kra02]. We use Krathwohl's revised version of Bloom's Taxonomy [Kra02], in order to assess the extent of educational objectives that can be reached by traditional means, and show how interactive visualizations can push these limits. The revised taxonomy distinguishes between the *Knowledge* dimension and the *Cognitive Processes* dimension, to better separate learning objectives. *Knowledge* refers to the content of a learning objective, whereas, the description of what is to be achieved is referred to as *Cognitive Process*. In the example sentence "students shall be able to remember the economics law of supply and demand" [Kra02], "remember" is the *Cognitive Process* and "economics law of supply and demand" is referred to as *Knowledge*.

Clustering algorithms, as all other algorithms, fall into the category of *Procedural Knowledge*. This type of knowledge is generally considered more complex than Factual Knowledge (e.g., number of states in the US) or Conceptual Knowledge (e.g., theory of evolution), and as such difficult to teach and learn. This points to the need for supportive learning and teaching tools.

Next, we list the cognitive processes we want to address with *EduClust* , ordered from simpler to more complex:

Remember: Retrieve relevant knowledge from long-term memory.
Understand: Determine the meaning of instructional messages, including oral, written, and graphic communication.
Apply: Carry out or use a procedure in a given situation.
Analyze: Break material into its constituent parts; detect how these relate to each other and to an overall structure or purpose.
Evaluate: Make judgments based on criteria and standards.
Create: Put elements together to form a novel, coherent whole or make an original product.

Knowledge transfer in traditional classroom settings often involves the cognitive processes of *Remember* and *Understand*. We designed *EduClust* in particular to also support the more complex cognitive processes. From these cognitive processes we derived concrete learning goals that informed the design process of *EduClust* . The identification of educational objectives or learning goals is important to improve the performance in classrooms [MM03, WM98]. Many learning goals for clustering algorithms are quite obvious, while we extracted others from the teaching literature [ES13, HK00] or publicly available course descriptions [oS17]. From the review of these descriptions we derived the following major learning goals in the context of clustering algorithms, together with the concrete cognitive processes involved (in *italics*). Each learning goal below is expressed as a concrete design consideration for *EduClust* .

Overall, the application should help students:

DC1: *Remember* what clustering algorithms exist, and recognize them based on their description.
DC2: *Remember* and *Understand* the different categories that the algorithms fall into.
DC3: *Understand* the meaning of the different algorithmic steps and parameters.
DC4: *Apply* the algorithms to data in a meaningful way.
DC5: *Analyze* the influence different distance measures and parameter settings have when applied to different kinds of data.
DC6: *Evaluate* which algorithms perform well (or not) in different settings, both with respect to the data and the parameters chosen.
DC7: *Create* an analysis scenario, i.e., a data set, where one algorithm is expected to outperform the others (as research suggests that adding custom datasets increases the pedagogical value to education platforms [SSMN04]).

We are aware that computer-based learning platforms can also have negative implications. As Northrup pointed out [Nor01], using computer-based teaching can reduce the contact with the educator, resulting in less feedback and time for questions and answers. However, *EduClust* is not meant to be solely an e-learning platform. It is designed to also be usable as a presentation tool during class and in close contact with a teacher.

4. The *EduClust* Application

We designed *EduClust* to take into account previous research in visualizing algorithms for teaching purposes (see Section 2.1) and the design considerations identified in Section 3. We decided to implement *EduClust* as a web-based prototype using D3 [BOH11] to make it widely accessible through a simple url (*anomyized due to review process*). We naturally support simultaneous users since calculations are performed client-side.

The online platform consists of five major components as seen in Fig. 1 and described next in detail. The components are tightly coupled to provide a cohesive education experience.

4.1. Selection Menu

The menu (Fig. 1 top) consists of three drop-down lists for choosing algorithms, datasets, or to load or create custom data.

Algorithms: We cover 9 common (**DC1, DC2**) clustering algorithms from three groups of methods [HK00]: partition-based linkage-based, and density-based clustering. While a good starting point for introductions to clustering, this set of algorithms is certainly not exhaustive. We plan to extend it further for the teaching of more advanced methods.

Datasets: We derived the initial pool of datasets from the literature [EKS*96, ABKS99] and our own experience with teaching and data analysis. The datasets contain various numbers of 2D clusters with different characteristics, such as shape or data point densities. As *EduClust* is meant to convey clusters in an easily comprehensible and visual way, we currently do not include the option to cluster 3D or higher-dimensional data as their visualizations add additional barriers to comprehension.

Custom Data: If users are not satisfied with this selection they manually add data points on the empty 2D plane. A custom dataset can be saved and then used with any clustering algorithm. This custom data functionality allows lecturers to react to student questions in the classroom and demonstrate the behavior of different algorithms on custom data on the fly. Custom data of under 600 2D-points can be clustered quickly on most clients and gives us a reasonable limit for the demonstration of our clustering algorithms.

With these Algorithm, Datasets, and Custom Data functionalities, students and teachers can benefit from experimenting with various data distributions and properties of clustering algorithms (**DC5, DC6**). They are also not restricted to just a few examples, but can build an infinite number of datasets to practice with or teach. Instead of having to implement entire algorithms to strengthen their understanding of clustering procedures (as suggested by Kann et al. [KLH97]), students can load and try to create 2D data distributions which are suitable only for specific algorithms (**DC7**), testing this way their understanding of the algorithmic procedures [SSMN04].

4.2. Parameter Settings

The dynamic Parameter Settings view (Fig. 1 left) represents the input parameters of the selected algorithm, as well as the chosen distance measure (**DC3**). We chose default values for each algorithm to target the best possible clustering results. The distance measure is initially set to the Euclidean distance since this measure is easy to understand for 2D projected data. The respective mathematical formula of the distance measure is also visible and provides more details. With this setup *EduClust* provides a flexible way to experiment with different clustering algorithms. Users can change the default input parameters as well as the distance measure to immediately see the effect of different settings (**DC5**) on the clustering results and the performance of the algorithm (**DC6**).

4.3. Cluster View

The visualizations implemented in *EduClust* (Fig. 1 center) are inspired by related work presented in Section 2.2 and 2.3. We extend existing work through a careful design of animated transitions between single steps of the algorithms (**DC3**) and auxiliary views to improve the understanding of specific clustering approaches (**DC3**) and to help evaluate the clustering performance (**DC6**).

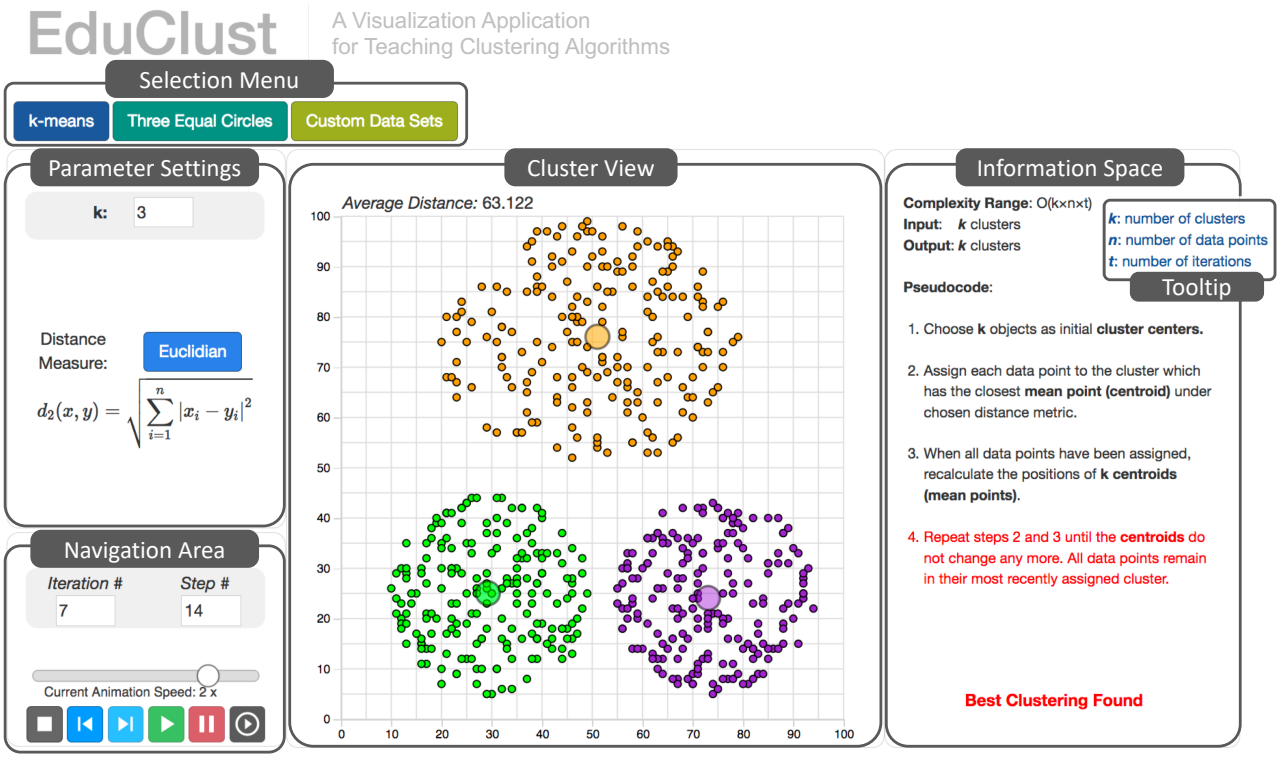

Figure 1: ***EduClust:*** *The visualization application consists of five main components: the selection menu for choosing an algorithm and dataset at the top; the parameter settings and navigation area for adjusting input parameters and controlling the algorithmic procedure to the left; the information space to get additional details and to provide the link to the textual pseudo code on the right; the cluster view to display the animated automatic clustering process in the center position. Depending on the selected algorithm additional visualizations are presented at the bottom. For each component, tooltips are used to provide further information.*

4.3.1. Scatterplot

We made the scatterplot the basis of the cluster view for three reasons: Scatterplots are well-known and easy to understand. They allow us to use the same representation for different algorithms and enable a simple brushing and linking functionality to connect data across auxiliary representations. Second, scatterplots help to easily understand the different cluster characteristics (i.e., shape, density, etc). For example many dots in a small region overplotting each other indicate a high density at this location. Additionally, convex data distributions can be easily distinguished from two-dimensional shape formations. Third, scatterplots are commonly used in data mining lectures and in the respective research papers [ABKS99, EKS*96]. Therefore, students and teachers can transfer existing examples from static learning material and display them in *EduClust*. We enriched the basic scatterplot visualization with specific extensions based on the underlying algorithm, like the minimum spanning tree for OPTICS, or additional visual cues to show the ε-distance while operating the DBSCAN algorithm, described next.

4.3.2. Dendrogram

For linkage-based methods an interactive dendrogram shows the hierarchical structure of the clustering. The dendrogram is directly linked to the scatterplot as shown in Fig. 2. Operations on the scatterplot (i. e., animation) are also visible in the dendrogram. Therefore, learners can easily follow the construction of the dendrogram to understand the hierarchical structure of the data. To define a cluster count, users drag a horizontal line on the dendrogram and see immediate feedback (**DC4**) in a color change of the resulting clusters in the scatterplot view. In Fig. 2 the user dragged the line just below the top node in the dendrogram resulting in one large and one small cluster.

We draw a small circle in the dendrogram where two clusters are joined. The circles of all underlying clusters share the same cluster color. Hovering over a circle highlights all sub-clusters in the dendrogram, as well as the corresponding data points in the scatterplot. This interactive exploration is meant to facilitate the understanding of the hierarchical clustering structure (**DC3**).

4.3.3. Reachability Plot

The output of the OPTICS algorithm is the reachability plot (Fig. 3), which represents each data point as two vertical bars in a histogram-like representation. The length of these bars encodes the reachability distance and the core distance separately. The data points are ordered according to their processing sequence. Clusters can be identified by searching for short bars expressing a low reachability and core distance (see similar colors at bottom of the histogram of image

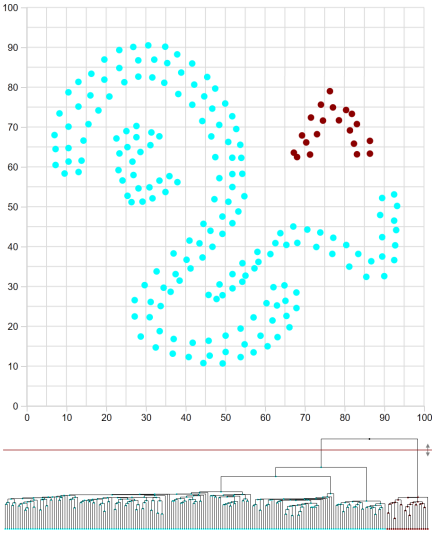

Figure 2: *Scatterplot (top) and dendrogram (bottom) for an example data set and single linkage clustering. The red line in the dendrogram indicates the cutting point for clustering and can be dragged up and down. Here the user chose a cutting point which results in two clusters. Points in the scatterplot and nodes in the dendrogram belonging to the same clusters are assigned the same color.*

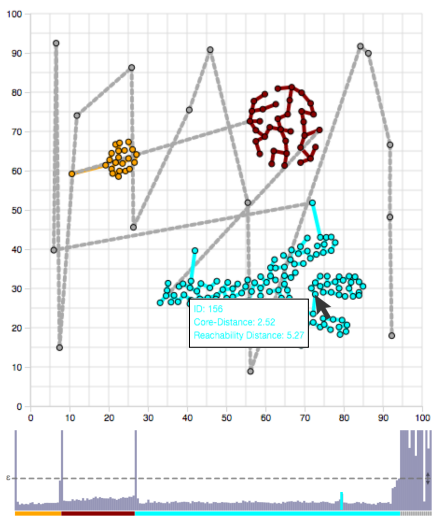

Figure 3: *The scatterplot and the reachability plot of the example data set with ε set to the default* 10*. Points and edges of the same cluster are filled with the same color. The hovered edge, connecting two clusters, has a reachability distance of* 5.27 *and is highlighted with the cluster color in the reachability plot. The horizontal dashed line in the reachability plot indicates the current value of the ε parameter.*

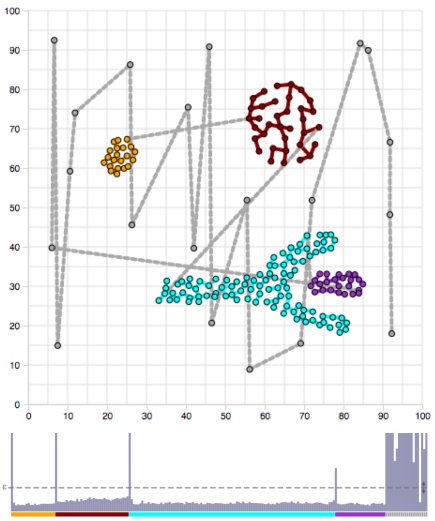

Figure 4: *The scatterplot and the reachability plot with the same data set as in Fig. 3 but with parameter ε set to* 5*. Here the former cyan cluster is split into the cyan and the purple ones. The parameter change led not only to a split of the cyan cluster at the desired edge, but the underlying spanning tree and the reachability plot were build up in another way resulting in different edges and bars.*

Fig. 3). To better understand the influence of the input parameter ε, a horizontal line in the plot indicates the current distance. This line adjusts automatically to the value of ε when the user changes the parameter. For a more direct manipulation, the user can simply drag the line to a position, which reflects the respective ε value they are interested in (**DC5**). The usage scenario in Section 5 explains the benefit of this visual feature.

4.4. Information Space

EduClust offers additional information on the algorithm chosen from the drop-down menu, (Fig. 1 right). This meta information includes the complexity of the algorithm, the required input parameters and the produced output (**DC1**). On hover, a tooltip gives further details on the meta data (Fig. 1 right).

Each step of the algorithm is detailed in the form of a pseudo-code-like step-wise description under the meta information (**DC3**). While the animation of the algorithm takes place, the respective part of the abstracted pseudo code is highlighted in red color. This functionality provides a direct link between the text description and the actual procedure visible in the cluster view (**DC3**). Study results and guidelines indicate that such a narration of the animation using dual coding of the content (i.e., text description and visual animation) helps to understand the underlying algorithm better [MA91, SBL93, HNS00].

4.5. Navigation Area

Like in a common media player, users can step through the automatic clustering using familiar icons and options (Fig. 1 left). The simplest way to experience the procedure of the algorithm is to press the "Play" button. This plays the animation of each step that the algorithm takes until it finishes (**DC3, DC4**). The animation speed can be adjusted using a horizontal slider and can be stopped at any time by clicking on the pause icon. Users can also see the algorithm step-by-step, and are able to go one step back and repeat certain iterations. Having single steps is a useful functionality to experience individual parts of the algorithm and to improve the understanding of the automatic procedure [SSMN04].

5. Usage Scenario

The following usage scenario is meant to show how both lecturers and students can benefit from *EduClust* when used in teaching.

In our imagined scenario, a lecturer uses *EduClust* to teach the OPTICS algorithm in class. She first introduces the basic idea and motivation behind the algorithm. She has prepared her slides by copying the text that describes the algorithmic steps from the Information Space panel in *EduClust* . Then, instead of showing several (previously prepared) examples on slides or static drawings on the black board, she opens *EduClust*, chooses a pre-defined example dataset from DBSCAN [EKS*96], and shows the students an animation of the algorithm with the default parameter settings ($\varepsilon = 10$ and $MinPoints = 3$); resulting in the cluster view shown in Fig. 3.

She now explains the importance of the spanning tree and hovers over individual edges to show the corresponding distances in the reachability plot. She uses the tooltips to relate the actual quantities with the lengths shown in both plots.

The lecturer next asks students for parameters that would allow to split the cyan cluster. She shows again the tooltip of the edge connecting the two components in Fig. 3. The shown reachability distance is 5.27 and she points out that the starting parameter for the ε distance needs to be lower than this value to separate the two components. Next, she asks students to propose a suitable lower bound for the ε distance. Students ask for the value of the second highest bar, which the lecturer reveals to be 3.47. In order not to split the cyan cluster into more than two new clusters, the value for ε, therefore, has to lie between 3.47–5.27. Students choose a value of $\varepsilon = 5$. The lecturer now reruns the OPTICS algorithm with the new ε value, which results in the new clustering shown in Fig. 4. Having saved the previous output to an image, the lecturer now shows both results in comparison. The students can see that the original cyan cluster is now split but that several, previously cyan, points are now classified as outliers.

After class, the lecturer works on additional revision slides for the students and exports the sequence of steps in the example to add them to the revision slide deck. As homework for the following week, the lecturer asks the students to compare the OPTICS algorithm against other clustering approaches. Their task is to come up with a data distribution that cannot be grouped adequately using other clustering algorithms. Since *EduClust* is available online, Ana, a student in the class accesses the software from home. She creates a custom data set similar to the example from class that she thinks might be suitable to solve the task. After saving the data, she applies different algorithms with different input parameters on her data set and checks the cluster behavior. Her first attempts fail, so she continues to edit the dataset until she has one that she can only cluster well using OPTICS. She exports the data in JSON format and sends her result to the lecturer, who can import the data and easily check for correctness.

6. Subjective Feedback from Classroom Use

One author of this paper used *EduClust* during the clustering lectures of a data mining course he taught. Our goal was to collect first feedback on *EduClust* . While *EduClust* is beneficial to both lecturers and students, we focus here on student feedback as the lecturer was too heavily involved in also building the tool. We do note, however, two features the lecturer noted as future improvements, specifically: the possibility to evaluate the cluster quality of the different approaches and to have a side-by-side comparison of multiple clustering outcomes.

Students who attended the respective lectures responded to a questionnaire on their use of the tool. They agreed (50%) or strongly agreed (50%) that integrating EduClust in the lecture helped them to understand the respective clustering algorithm taught. Moreover, 53% agreed or strongly agreed (47%) that they would use the tool to prepare for the exam. When checking our server logs between the lecture about clustering and the final exam, we identified 13 different IP accesses on average per day with a peak of 44 different IP addresses on a single day (48 students in class). Thus, our logs indicate students did indeed use the application to prepare for the exam. After using the application for studying, students were asked whether the current implementation of the different algorithms helped them to better understand the algorithmic behavior. 22 students agreed or strongly agreed (77%) with this statement, five students (18%) were neutral in their decision and one student (4%) disagreed. An overview of their feedback is given in Table 2.

Table 2: *Likert scale results to the question "The current implementation of <name of the algorithm> helped me to better understand the algorithmic behavior."*

Clustering Algorithm	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
K-Means	0	3.6%	17.9%	53.6%	25%
ISODATA	0	8%	40%	24%	28%
Single Linkage	0	3.7%	3.7%	59.3%	33.3%
Average Linkage	0	7.4%	14.8%	51.9%	25.9%
DBSCAN	0	0	11.1%	55.6%	33.3%
OPTICS	0	3.7%	22.2%	51.9%	22.2%

7. Conclusion

EduClust is an online visualization application that combines interactive visualizations of clustering algorithms to support lecturers and students in active learning scenarios. We designed *EduClust* based on educational principles covering also complex cognitive processes like apply, analyze, and evaluate. With *EduClust* we offer lecturers the means to extend their current practices in teaching clustering algorithms and to quickly prepare a large set of clustering examples. We tested *EduClust* in the classroom. Not only did the students value the *usefulness* of *EduClust*, they all stated that they would use the software to prepare for the final exam, a statement that seems to be supported by the increase in server accesses during the preparation phase for the exam. Thus, their statement reflects true intentions rather than a possible effect of desirability bias.

Adding new algorithms and visualizations in *EduClust* currently requires understanding some intricacies in the software's architecture. We plan to extend the application into a platform that allows contributors to upload new algorithms and visualizations. We also plan to better integrate *EduClust* into classroom workflows, for example by recording and providing statistics about interactions (or mistakes in the use of parameters) in order to inform both students and teachers of possible problematic areas to focus on.

References

[ABKS99] ANKERST M., BREUNIG M. M., KRIEGEL H.-P., SANDER J.: Optics: ordering points to identify the clustering structure. In *ACM Sigmod record* (1999), vol. 28, ACM, pp. 49–60. 2, 3, 4, 5

[AH13] ALTARAWNEH R., HUMAYOUN S. R.: A two-perspective visualization approach for utilizing visualization power in computer science education. In *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research* (The Netherlands, 2013), CSERC '13, Open Universiteit, Heerlen, pp. 8:85–8:91. 2

[BCD*09] BERTHOLD M. R., CEBRON N., DILL F., GABRIEL T. R., KÖTTER T., MEINL T., OHL P., THIEL K., WISWEDEL B.: Knime-the konstanz information miner: version 2.0 and beyond. *AcM SIGKDD explorations Newsletter 11*, 1 (2009), 26–31. 2

[BEF*56] BLOOM B. S., ENGELHART M., FURST E., HILL W., KRATHWOHL D.: Taxonomy of educational objectives. vol. 1: Cognitive domain. *New York: McKay* (1956), 20–24. 3

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (Dec. 2011), 2301–2309. 4

[Bos18] BOSTOCK M.:. Website, 2018. https://beta.observablehq.com/collection/explorables, Retrieved March 2018. 1, 2

[Bre99] BREUNIG M.: Corelevel-reachability viewer, 1999. http://www.dbs.informatik.uni-muenchen.de/Forschung/KDD/Clustering/OPTICS/Demo/. Retrieved March 2018. 3

[CCL*16] CHEN Q., CHEN Y., LIU D., SHI C., WU Y., QU H.: Peakvizor: Visual analytics of peaks in video clickstreams from massive open online courses. *IEEE Transactions on Visualization and Computer Graphics 22*, 10 (Oct 2016), 2315–2330. 2

[CCZ*16] CHEN Y., CHEN Q., ZHAO M., BOYER S., VEERAMACHANENI K., QU H.: Dropoutseer: Visualizing learning patterns in massive open online courses for dropout reasoning and prediction. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)* (Oct 2016), pp. 111–120. 2

[EKS*96] ESTER M., KRIEGEL H.-P., SANDER J., XU X., ET AL.: A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231. 2, 3, 4, 5, 7

[ES13] ESTER M., SANDER J.: *Knowledge discovery in databases: Techniken und Anwendungen*. Springer-Verlag, 2013. 4

[FHH*09] FRANK E., HALL M., HOLMES G., KIRKBY R., PFAHRINGER B., WITTEN I. H., TRIGG L.: Weka-a machine learning workbench for data mining. In *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 1269–1277. 2

[GASGTSGP10] GOMEZ AGUILAR D. A., SUAREZ GUERRERO C., THERON SANCHEZ R., GARCIA PENALVO F.: Visual analytics to support e-learning. In *Advances in Learning Processes*. InTech, 2010. 2

[GR69] GOWER J. C., ROSS G.: Minimum spanning trees and single linkage cluster analysis. *Applied statistics* (1969), 54–64. 2

[Har14] HARRIS N.: K-means visualization using a scatterplot with dividing planes, 2014. https://www.naftaliharris.com/blog/visualizing-k-means-clustering/. Retrieved March 2018. 2

[Har15] HARRIS N.: Visualizing dbscan clustering, 2015. https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/. Retrieved March 2018. 3

[HDS02] HUNDHAUSEN C. D., DOUGLAS S. A., STASKO J. T.: A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages & Computing 13*, 3 (2002), 259 – 290. 2

[HK*98] HINNEBURG A., KEIM D. A., ET AL.: An efficient approach to clustering in large multimedia databases with noise. In *KDD* (1998), vol. 98, pp. 58–65. 3

[HK00] HAN J., KAMBER M.: *Data mining: concepts and techniques*. Morgan Kaufmann, 2000. 4

[HKLH12] HALIM S., KOH Z. C., LOH V. B. H., HALIM F.: Learning algorithms with unified and interactive web-based visualization. *Olympiads in Informatics 6* (2012), 53–68. 2

[HNS00] HANSEN S. R., NARAYANAN N. H., SCHRIMPSHER D.: Helping learners visualize and comprehend algorithms. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning 2*, 1 (2000), 10. 2, 6

[Inc] INC. P. T.: Dendrograms in python. https://plot.ly/python/dendrogram/. Retrieved March 2018. 2

[Inc89] INC. S. I.: jmp statistical discovery, 1989. https://www.jmp.com. Retrieved March 2018. 2

[KLH97] KANN C., LINDEMAN R. W., HELLER R.: Integrating algorithm animation into a learning environment. *Computers & Education 28*, 4 (1997), 223–228. 4

[Kra02] KRATHWOHL D. R.: A revision of bloom's taxonomy: An overview. *Theory into practice 41*, 4 (2002), 212–218. 1, 3

[Kun16] KUNIN D.: Seeing theorie, 2016. http://students.brown.edu/seeing-theory/. Retrieved March 2018. 2

[LBS94] LAWRENCE A. W., BADRE A. M., STASKO J. T.: Empirically evaluating the use of animations to teach algorithms. In *Visual Languages, 1994. Proceedings., IEEE Symposium on* (1994), IEEE, pp. 48–54. 2

[MA91] MAYER R. E., ANDERSON R. B.: Animations need narrations: An experimental test of a dual-coding hypothesis. *Journal of educational psychology 83*, 4 (1991), 484. 6

[Mac67] MACQUEEN J.: Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (Berkeley, Calif., 1967), University of California Press, pp. 281–297. 2

[MM03] MARZANO R. J., MARZANO J. S.: The key to classroom management. *Educational Leadership 61*, 1 (2003), 6–13. 4

[Moh18] MOHAN K.: K-means visualization using a scatterplot, 2018. http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html. Retrieved March 2018. 2

[Nor01] NORTHRUP P.: A framework for designing interactivity into web-based instruction. *Educational Technology 41*, 2 (2001), 31–39. 4

[oS17] OF SCIENCE P. E. C.: Lesson 10: Clustering, 2017. https://onlinecourses.science.psu.edu/stat555/node/11. Retrieved March 2018. 4

[R D08] R DEVELOPMENT CORE TEAM: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0. 2

[Sai13] SAITA K.: K-means visualization using a scatterplot with connections, 2013. http://tech.nitoyon.com/en/blog/2013/11/07/k-means/. Retrieved March 2018. 2

[SBL93] STASKO J., BADRE A., LEWIS C.: Do algorithm animations assist learning?: An empirical study and analysis. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (New York, NY, USA, 1993), CHI '93, ACM, pp. 61–66. 6

[SCA*10] SHAFFER C. A., COOPER M. L., ALON A. J. D., AKBAR M., STEWART M., PONCE S., EDWARDS S. H.: Algorithm visualization: The state of the field. *ACM Transactions on Computing Education (TOCE) 10*, 3 (2010), 9. 2

[SF11] STEVEN H., FELIX H.: Visualgo, 2011. https://visualgo.net/. Retrieved March 2018. 2

[SHFB17] SIEGER T., HURLEY C. B., FIŠER K., BELEITES C.: Interactive dendrograms: The r packages idendro and idendr0. *Journal of Statistical Software 76*, 1 (2017), 1–22. 2

[Sim97] SIMS R.: Interactive learning as an" emerging" technology: A reassessment of interactive and instructional design strategies. *Australasian Journal of Educational Technology 13*, 1 (1997). 2

[SSMN04] SARAIYA P., SHAFFER C. A., MCCRICKARD D. S., NORTH C.: Effective features of algorithm visualizations. *SIGCSE Bull. 36*, 1 (Mar. 2004), 382–386. 4, 6

[SST*17] SCHWAB M., STROBELT H., TOMPKIN J., FREDERICKS C., HUFF C., HIGGINS D., STREZHNEV A., KOMISARCHIK M., KING G., PFISTER H.: booc. io: An education system with hierarchical concept maps and dynamic non-linear learning plans. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (2017), 571–580. 2

[Vic18] VICTOR B.: Explorable explanations, Retrieved March 2018. http://worrydream.com/ExplorableExplanations/. 1, 2

[Wik17] WIKIPEDIA: Optics algorithm — Wikipedia, the free encyclopedia, 2017. https://en.wikipedia.org/wiki/OPTICS_algorithm. Retrieved March 2018. 3

[WM98] WIGGINS G., MCTIGHE J.: Understanding by design. 4

[WYD*16] WU T., YAO Y., DUAN Y., FAN X., QU H.: Networkseer: Visual analysis for social network in moocs. In *2016 IEEE Pacific Visualization Symposium (PacificVis)* (April 2016), pp. 194–198. 2